1. Who is the father of C language?
a) Steve Jobs
b) James Gosling
c) Dennis Ritchie
d) Rasmus Lerdorf

ANSWER:- -: c
Explanation: Dennis Ritchie is the father of C Programming Language. C programming language was developed in 1972 at American Telephone & Telegraph Bell Laboratories of USA.

2. Which of the following is not a valid C variable name?
a) int number;
b) float rate;
c) int variable_count;
d) int $main;

ANSWER:- -: d
Explanation: Since only underscore and no other special character is allowed in a variable name, it results in an error.

3. All keywords in C are in _____
a) LowerCase letters
b) UpperCase letters
c) CamelCase letters
d) None of the mentioned

ANSWER:- -: a
Explanation: None.

4. Which of the following is true for variable names in C?
a) They can contain alphanumeric characters as well as special characters
b) It is not an error to declare a variable to be one of the keywords(like goto, static)
c) Variable names cannot start with a digit
d) Variable can be of any length

ANSWER:- -: c
Explanation: According to the syntax for C variable name, it cannot start with a digit.

5. Which is valid C expression?
a) int my_num = 100,000;
b) int my_num = 100000;
c) int my num = 1000;
d) int $my_num = 10000;

ANSWER:- -: b
Explanation: Space, comma and $ cannot be used in a variable name.

6. Which of the following cannot be a variable name in C?
a) volatile
b) true
c) friend
d) export

ANSWER:- -: a
Explanation: volatile is C keyword.

7. What is short int in C programming?
a) The basic data type of C
b) Qualifier
c) Short is the qualifier and int is the basic data type
d) All of the mentioned

ANSWER:- -: c
Explanation: None.

8. Which of the following declaration is not supported by C language?
a) String str;
b) char *str;
c) float str = 3e2;
d) Both “String str;” and “float str = 3e2;”

ANSWER:- -: a
Explanation: It is legal in Java, but not in C language.

9. Which keyword is used to prevent any changes in the variable within a C program?
a) immutable
b) mutable
c) const
d) volatile

ANSWER:- -: c
Explanation: const is a keyword constant in C program.

10. What is the result of logical or relational expression in C?
a) True or False
b) 0 or 1
c) 0 if an expression is false and any positive number if an expression is true
d) None of the mentioned

ANSWER:- -: b
Explanation: None.

11. Which of the following typecasting is accepted by C language?
a) Widening conversions
b) Narrowing conversions
c) Widening & Narrowing conversions
d) None of the mentioned

ANSWER:- -: c
Explanation: None.

12. Where in C the order of precedence of operators do not exist?
a) Within conditional statements, if, else
b) Within while, do-while
c) Within a macro definition
d) None of the mentioned

ANSWER:- -: d
Explanation: None.

13. Which of the following is NOT possible with any 2 operators in C?
a) Different precedence, same associativity
b) Different precedence, different associativity
c) Same precedence, different associativity
d) All of the mentioned

ANSWER:- -: c
Explanation: None.

14. What is an example of iteration in C?
a) for
b) while
c) do-while
d) all of the mentioned

ANSWER:- -: d
Explanation: None.

15. Functions can return enumeration constants in C?
a) true
b) false
c) depends on the compiler
d) depends on the standard

ANSWER:- -: a
Explanation: None.

16. Functions in C Language are always _____
a) Internal
b) External
c) Both Internal and External
d) External and Internal are not valid terms for functions

ANSWER:- -: b
Explanation: None.

17. Which of following is not accepted in C?
a) static a = 10; //static as
b) static int func (int); //parameter as static
c) static static int a; //a static variable prefixed with static
d) all of the mentioned

ANSWER:- -: c
Explanation: None.

18. Property which allows to produce different executable for different platforms in C is called?
a) File inclusion
b) Selective inclusion
c) Conditional compilation
d) Recursive macros

ANSWER:- -: c
Explanation: Conditional compilation is the preprocessor facility to produce a different executable.

19. What is #include <stdio.h>?
a) Preprocessor directive
b) Inclusion directive
c) File inclusion directive
d) None of the mentioned

ANSWER:- -: a
Explanation: None.

20. C preprocessors can have compiler specific features.
a) True
b) False
c) Depends on the standard
d) Depends on the platform

ANSWER:- -: a
Explanation: #pragma is compiler specific feature.

21. Which of the following are C preprocessors?
a) #ifdef
b) #define
c) #endif
d) all of the mentioned

ANSWER:- -: d
Explanation: None.

22. The C-preprocessors are specified with _____ symbol.
a) #
b) $
c) ” ”
d) &

ANSWER:- -: a
Explanation: The C-preprocessors are specified with # symbol.

23. How is search done in #include and #include “somelibrary.h” according to C standard?
a) When former is used, current directory is

searched and when latter is used, standard directory is searched
b) When former is used, standard directory is searched and when latter is used, current directory is searched
c) When former is used, search is done in implementation defined manner and when latter is used, current directory is searched
d) For both, search for 'somelibrary' is done in implementation-defined places

ANSWER:- -: b
Explanation: None.

24. How many number of pointer (*) does C have against a pointer variable declaration?
a) 7
b) 127
c) 255
d) No limits

ANSWER:- -: d
Explanation: None.

25. Which of the following is not possible statically in C language?
a) Jagged Array
b) Rectangular Array
c) Cuboidal Array
d) Multidimensional Array

ANSWER:- -: a
Explanation: None.

26. Which of the following return-type cannot be used for a function in C?
a) char *
b) struct
c) void
d) none of the mentioned

ANSWER:- -: d
Explanation: None.

27. The standard header _____ is used for variable list arguments (...) in C.
a) <stdio.h >
b) <stdlib.h>
c) <math.h>
d) <stdarg.h>

ANSWER:- -: d
Explanation: None.

28. When a C program is started, O.S environment is responsible for opening file and providing pointer for that file?
a) Standard input
b) Standard output
c) Standard error
d) All of the mentioned

ANSWER:- -: d
Explanation: None.

29. In C language, FILE is of which data type?
a) int
b) char *
c) struct
d) None of the mentioned

ANSWER:- -: c
Explanation: None.

30. What is the sizeof(char) in a 32-bit C compiler?
a) 1 bit
b) 2 bits
c) 1 Byte
d) 2 Bytes

ANSWER:- -: c
Explanation: None.

31. Which of the following is not an operator in C?
a) ,
b) sizeof()
c) ~
d) None of the mentioned

ANSWER:- -: d
Explanation: None.

32. scanf() is a predefined function in_____header file.
a) stdlib. h
b) ctype. h
c) stdio. h
d) stdarg. h

ANSWER:- -: c
Explanation: scanf() is a predefined function in "stdio.h" header file.printf and scanf() carry out input and output functions in C. These functions statements are present in the header file stdio.h.

33. What is meant by 'a' in the following C operation?

```
fp = fopen("Random.txt", "a");
```

a) Attach
b) Append
c) Apprehend
d) Add

ANSWER:- -: b
Explanation: None.

34. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int y = 10000;
    int y = 34;
    printf("Hello World! %d\n", y);
    return 0;
}
```

a) Compile time error
b) Hello World! 34
c) Hello World! 1000
d) Hello World! followed by a junk value

ANSWER:- -: a
Explanation: Since y is already defined, redefining it results in an error.
Output:
$ cc pgm2.c
pgm2.c: In function 'main':
pgm2.c:5: error: redefinition of 'y'
pgm2.c:4: note: previous definition of 'y' was here

35. What will happen if the following C code is executed?

```
#include <stdio.h>
int main()
{
    int main = 3;
    printf("%d", main);
    return 0;
}
```

a) It will cause a compile-time error
b) It will cause a run-time error
c) It will run without any error and prints 3
d) It will experience infinite looping

ANSWER:- -: c
Explanation: A C program can have same function name and same variable name.
$ cc pgm3.c
$ a.out
3

36. What will be the output of the following C code?

```
#include  <stdio.h>
int main()
{
   signed char chr;
   chr = 128;
   printf("%d\n", chr);
   return 0;
}
```

a) 128
b) -128
c) Depends on the compiler
d) None of the mentioned

ANSWER:- -: b
Explanation: The range of signed character is from -128 to +127. Since we are assigning a value of 128 to the variable 'chr', the result will be negative. 128 in binary is represented as "1000 0000" for character datatype. As you can see that the sign bit is set to 1, followed by 7 zeros (0), its final decimal value will be -128 (negative 128).
Output:
$ cc pgm2.c
$ a.out
-128

37. What will be the output of the following C code on a 64 bit machine?

```
#include <stdio.h>
union Sti
{
    int nu;
    char m;
};
int main()
{
    union Sti s;
     printf("%d", sizeof(s));
     return 0;
 }
```

a) 8
b) 5
c) 9
d) 4

ANSWER:- -: d
Explanation: Since the size of a union is the size of its maximum data type, here int is the largest data type. Hence the size of the union is 4.
Output:
$ cc pgm7.c
$ a.out
4

38. What will be the output of the following C function?

```
1.    #include <stdio.h>
2.    enum birds {SPARROW, PEACOCK,
   PARROT};
3.    enum animals {TIGER = 8,
   LION, RABBIT, ZEBRA};
4.    int main()
5.    {
6.        enum birds m = TIGER;
7.        int k;
8.        k = m;
9.        printf("%d\n", k);
10.        return 0;
11.    }
```

a) 0
b) Compile time error
c) 1
d) 8

ANSWER:- -: d
Explanation: m is an integer constant, hence it is compatible.
Output:
$ cc pgm5.c
$ a.out
8

39. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int const print()
3.    {
4.        printf("Sanfoundry.com");
5.        return 0;
6.    }
7.    void main()
8.    {
9.        print();
10.    }
```

a) Error because function name cannot be preceded by const
b) Sanfoundry.com
c) Sanfoundry.com is printed infinite times
d) Blank screen, no output

ANSWER:- -: b
Explanation: None.
Output:
$ cc pgm13.c
$ a.out
Sanfoundry.com

40. Will the following C code compile without any error?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        for (int k = 0; k < 10;
   k++);
5.            return 0;
6.    }
```

a) Yes
b) No
c) Depends on the C standard implemented by compilers
d) Error

ANSWER:- -: c
Explanation: Compilers implementing C90 do not allow this, but compilers implementing C99 allow it.
Output:
$ cc pgm4.c
pgm4.c: In function ‘main’:
pgm4.c:4: error: ‘for’ loop initial declarations are only allowed in C99 mode
pgm4.c:4: note: use option -std=c99 or -std=gnu99 to compile your code

41. What will be the final value of x in the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int x = 5 * 9 / 3 + 9;
5.    }
```

a) 3.75
b) Depends on compiler
c) 24
d) 3

ANSWER:- -: c
Explanation: None.

42. What will be the output of the following C code? (Initial values: x= 7, y = 8)

```
#include <stdio.h>
void main()
{
    float x;
    int y;
    printf("enter two numbers \n");
    scanf("%f %f", &x, &y);
    printf("%f, %d", x, y);
}
```

a) 7.000000, 7
b) Run time error
c) 7.000000, junk
d) Varies

ANSWER:- -: c
Explanation: None.

43. What will be the output of the following C code considering the size of a short int is 2, char is 1 and int is 4 bytes?

```
#include <stdio.h>
int main()
{
    short int i = 20;
    char c = 97;
    printf("%d, %d, %d\n", sizeof(i), sizeof(c), sizeof(c + i));
    return 0;
}
```

a) 2, 1, 2
b) 2, 1, 1
c) 2, 1, 4
d) 2, 2, 8

ANSWER:- -: c
Explanation: None.

44. What is the difference between the following 2 C codes?

```
#include <stdio.h> //Program 1
int main()
{
    int d, a = 1, b = 2;
    d =  a++ + ++b;
    printf("%d %d %d", d, a, b);
}
```

```
#include <stdio.h> //Program 2
int main()
{
    int d, a = 1, b = 2;
    d =  a++ +++b;
    printf("%d %d %d", d, a, b);
}
```

a) No difference as space doesn't make any difference, values of a, b, d are same in both the case
b) Space does make a difference, values of a, b, d are different
c) Program 1 has syntax error, program 2 is not
d) Program 2 has syntax error, program 1 is not

ANSWER:- -: d
Explanation: None.

45. What will be the output of the following C code snippet?

```
#include <stdio.h>
void main()
{
    1 < 2 ? return 1: return 2;
}
```

a) returns 1
b) returns 2
c) Varies
d) Compile time error

ANSWER:- -: d
Explanation: None.

46. What will be the value of the following assignment expression?

(x = foo())!= 1 considering foo() returns 2

a) 2
b) True
c) 1
d) 0

ANSWER:- -: a
Explanation: None.

47. What will be the output of the following C function?

```
#include <stdio.h>
void reverse(int i);
int main()
{
    reverse(1);
}
void reverse(int i)
{
    if (i > 5)
        return ;
     printf("%d ", i);
     return reverse((i++, i));
 }
```

a) 1 2 3 4 5
b) Segmentation fault
c) Compilation error
d) Undefined behaviour

ANSWER:- -: a
Explanation: None.

48. What will be the final values of i and j in the following C code?

```
#include <stdio.h>
int x = 0;
int f()
{
    if (x == 0)
        return x + 1;
    else
        return x - 1;
}
int g()
{
    return x++;
}
int main()
{
    int i = (f() + g()) | g(); //bitwise or
    int j = g() | (f() + g()); //bitwise or
}
```

a) i value is 1 and j value is 1
b) i value is 0 and j value is 0
c) i value is 1 and j value is undefined
d) i and j value are undefined

ANSWER:- -: c
Explanation: None.

49. Comment on the following C statement.

```
n = 1;
printf("%d, %dn", 3*n, n++);
```

a) Output will be 3, 2
b) Output will be 3, 1
c) Output will be 6, 1
d) Output is compiler dependent

ANSWER:- -: d
Explanation: None.

50. How many times i value is checked in the following C program?

```
#include <stdio.h>
int main()
{
    int i = 0;
    while (i < 3)
        i++;
    printf("In while loop\n");
}
```

a) 2
b) 3
c) 4
d) 1

ANSWER:- -: c
Explanation: None.

51. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.    int main()
3.    {
4.        int i = 0;
5.        do
6.        {
7.            i++;
8.            if (i == 2)
9.                continue;
10.                printf("In
   while loop ");
11.         } while (i < 2);
12.         printf("%d\n", i);
13.    }
```

a) In while loop 2
b) In while loop in while loop 3
c) In while loop 3
d) Infinite loop

ANSWER:- -: a
Explanation: None.

52. What will be the data type returned for the following C function?

```
1.    #include <stdio.h>
2.    int func()
3.    {
4.        return (double)(char)5.0;
5.    }
```

a) char
b) int
c) double
d) multiple type-casting in return is illegal

ANSWER:- -: b
Explanation: None.

53. What is the problem in the following C declarations?

```
int func(int);
double func(int);
int func(float);
```

a) A function with same name cannot have different signatures
b) A function with same name cannot have different return types
c) A function with same name cannot have different number of parameters
d) All of the mentioned

ANSWER:- -: d
Explanation: None.

54. Which option should be selected to work the following C expression?

```
string p = "HELLO";
```

a) typedef char [] string;
b) typedef char *string;
c) typedef char [] string; and typedef char *string;
d) Such expression cannot be generated in C

ANSWER:- -: b
Explanation: None.

55. What is the meaning of the following C statement?

printf("%10s", state);

a) 10 spaces before the string state is printed
b) Print empty spaces if the string state is less than 10 characters
c) Print the last 10 characters of the string
d) None of the mentioned

ANSWER:- -: b
Explanation: None.

56. What are the elements present in the array of the following C code?

```
int array[5] = {5};
```

a) 5, 5, 5, 5, 5
b) 5, 0, 0, 0, 0
c) 5, (garbage), (garbage), (garbage), (garbage)
d) (garbage), (garbage), (garbage), (garbage), 5

ANSWER:- -: b
Explanation: None.

57. What will be the output of the following C function when EOF returns?

```
int fputs(char *line, FILE *fp)
```

a) '' character of array line is encountered
b) 'n' character in array line is encountered
c) 't' character in array line is encountered
d) When an error occurs

ANSWER:- -: d
Explanation: None.

58. Which part of the program address space is p stored in the following C code?

```
#include <stdio.h>
int *p;
int main()
{
    int i = 0;
    p = &i;
    return 0;
}
```

a) Code/text segment
b) Data segment
c) Bss segment
d) Stack

ANSWER:- -: c
Explanation: None.

59. Which of the following sequences are unaccepted in C language?
a)

```
#if
#else
#endif
```

b)

```
#if
#elif
#endif
```

c)

```
#if
#if
#endif
```

d)

```
#if
#undef
#endif
```

ANSWER:- -: c
Explanation: None.

59. Comment on the output of following C code.

```
#include <stdio.h>
main()
{
    char *p = 0;
    *p = 'a';
    printf("value in pointer p is %c\n", *p);
}
```

a) It will print a
b) It will print 0
c) Compile time error
d) Run time error

ANSWER:- -: d
Output:
$ cc pgm.c
$ a.out
Segmentation fault (core dumped)

60. What is the output of this C code?

```
#include <stdio.h>
main()
{
    if (sizeof(int) > -1)
        printf("True");
    else
        printf("False");
}
```

a) True
b) False

ANSWER:- -: b
Output:
$ cc pgm.c
$ a.out
False

61. What is the output of this C code?

```
#include <stdio.h>
```

```
2.    main()
3.    {
4.        char *p = "Sanfoundry C-
    Test";
5.        p[0] = 'a';
6.        p[1] = 'b';
7.        printf("%s", p);
8.    }
```

a) abnfoundry C-Test
b) Sanfoundry C-Test
c) Compile time error
d) Run time error

ANSWER:- -: d
Output:
$ cc pgm.c
$ a.out
Segmentation fault (core dumped)

62. What is the output of this C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        float f = 0.1;
5.        if (f == 0.1)
6.            printf("True");
7.        else
8.            printf("False");
9.    }
```

a) True
b) False

ANSWER:- -: b
Output:
$ cc pgm.c
$ a.out
False

63. What is the output of this C code?

```
1.    #include <stdio.h>
2.    main()
3.    {
4.        int n = 0, m = 0;
5.        if (n > 0)
6.            if (m > 0)
7.                printf("True");
8.        else
9.            printf("False");
10.   }
```

a) True
b) False
c) No Output will be printed
d) Run Time Error

ANSWER:- -: c
Output:
$ cc pgm.c
$ a.out
$

1. C99 standard guarantees uniqueness of _____ characters for internal names.
a) 31
b) 63
c) 12
d) 14

ANSWER:- b
Explanation: ISO C99 compiler may consider only first 63 characters for internal names.

2. C99 standard guarantees uniqueness of _____ characters for external names.
a) 31
b) 6
c) 12
d) 14

ANSWER:- a
Explanation: ISO C99 compiler may consider only first 31 characters for external names.

3. Which of the following is not a valid variable name declaration?
a) int __a3;
b) int __3a;
c) int __A3;
d) None of the mentioned

ANSWER:- d
Explanation: None.

4. Which of the following is not a valid variable name declaration?
a) int _a3;
b) int a_3;
c) int 3_a;
d) int _3a

ANSWER:- c
Explanation: Variable name cannot start with a digit.

5. Why do variable names beginning with the underscore is not encouraged?
a) It is not standardized
b) To avoid conflicts since assemblers and loaders use such names
c) To avoid conflicts since library routines use such names
d) To avoid conflicts with environment variables of an operating system

ANSWER:- c
Explanation: None.

6. All keywords in C are in _____
a) LowerCase letters
b) UpperCase letters
c) CamelCase letters
d) None of the mentioned

ANSWER:- a
Explanation: None.

7. Variable name resolution (number of significant characters for the uniqueness of variable) depends on _____
a) Compiler and linker implementations
b) Assemblers and loaders implementations
c) C language
d) None of the mentioned

ANSWER:- a
Explanation: It depends on the standard to which compiler and linkers are adhering.

8. Which of the following is not a valid C variable name?
a) int number;
b) float rate;
c) int variable_count;
d) int $main;

ANSWER:- d
Explanation: Since only underscore and no other special character is allowed in a variable name, it results in an error

9. Which of the following is true for variable names in C?
a) They can contain alphanumeric characters as well as special characters
b) It is not an error to declare a variable to be one of the keywords(like goto, static)
c) Variable names cannot start with a digit
d) Variable can be of any length

ANSWER:- c
Explanation: According to the syntax for C variable name, it cannot start with a digit.

1. Which is valid C expression?
a) int my_num = 100,000;
b) int my_num = 100000;
c) int my num = 1000;
d) int $my_num = 10000;

ANSWER:- b
Explanation: Space, comma and $ cannot be used in a variable name.

2. What will be the output of the following C code?

Participate Now!

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        printf("Hello World! %d
   \n", x);
5.        return 0;
6.    }
```

a) Hello World! x;
b) Hello World! followed by a junk value
c) Compile time error
d) Hello World!

ANSWER:- c
Explanation: It results in an error since x is used without declaring the variable x.
Output:
$ cc pgm1.c
pgm1.c: In function ‘main’:
pgm1.c:4: error: ‘x’ undeclared (first use in this function)
pgm1.c:4: error: (Each undeclared identifier is reported only once
pgm1.c:4: error: for each function it appears in.)

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int y = 10000;
5.        int y = 34;
6.        printf("Hello
   World! %d\n", y);
```

```
7.          return 0;
8.      }
```

a) Compile time error
b) Hello World! 34
c) Hello World! 1000
d) Hello World! followed by a junk value

ANSWER:- a
Explanation: Since y is already defined, redefining it results in an error.
Output:
$ cc pgm2.c
pgm2.c: In function ‘main’:
pgm2.c:5: error: redefinition of ‘y’
pgm2.c:4: note: previous definition of ‘y’ was here

4. Which of the following is not a valid variable name declaration?
a) float PI = 3.14;
b) double PI = 3.14;
c) int PI = 3.14;
d) #define PI 3.14

ANSWER:- d
Explanation: #define PI 3.14 is a macro preprocessor, it is a textual substitution.

5. What will happen if the following C code is executed?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int main = 3;
5.          printf("%d", main);
6.          return 0;
7.      }
```

a) It will cause a compile-time error
b) It will cause a run-time error
c) It will run without any error and prints 3
d) It will experience infinite looping

ANSWER:- c
Explanation: A C program can have same function name and same variable name.
$ cc pgm3.c
$ a.out
3

6. What is the problem in the following variable declaration?

float 3Bedroom-Hall-Kitchen?;

a) The variable name begins with an integer
b) The special character ‘-‘
c) The special character ‘?’
d) All of the mentioned

ANSWER:- d
Explanation: A variable name cannot start with an integer, along with that the C compiler interprets the ‘-‘ and ‘?’ as a minus operator and a question mark operator respectively.

7. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int ThisIsVariableName =
    12;
5.          int ThisIsVariablename =
    14;
6.          printf("%d",
    ThisIsVariablename);
7.          return 0;
8.      }
```

a) The program will print 12
b) The program will print 14
c) The program will have a runtime error
d) The program will cause a compile-time error due to redeclaration

ANSWER:- b
Explanation: Variable names ThisIsVariablename and ThisIsVariableName are both distinct as C is case sensitive.
Output:
$ cc pgm4.c
$ a.out
14

8. Which of the following cannot be a variable name in C?
a) volatile
b) true
c) friend
d) export

ANSWER:- a
Explanation: volatile is C keyword.

1. What will be the output of the following C code?

```
1.      #include <stdio.h>
```

```
int main()
{
    int a[5] = {1, 2, 3, 4, 5};
    int i;
    for (i = 0; i < 5; i++)
        if ((char)a[i] == '5')
            printf("%d\n", a[i]);
        else
            printf("FAIL\n");
}
```

a) The compiler will flag an error
b) The program will compile and print the output 5
c) The program will compile and print the ASCII value of 5
d) The program will compile and print FAIL for 5 times

ANSWER:- d
Explanation: The ASCII value of 5 is 53, the char type-casted integral value 5 is 5 only.
Output:
$ cc pgm1.c
$ a.out
FAIL
FAIL
FAIL
FAIL
FAIL

2. The format identifier '%i' is also used for ____ data type.
a) char
b) int
c) float
d) double

ANSWER:- b
Explanation: Both %d and %i can be used as a format identifier for int data type.

3. Which data type is most suitable for storing a number 65000 in a 32-bit system?
a) signed short
b) unsigned short
c) long
d) int

ANSWER:- b
Explanation: 65000 comes in the range of short (16-bit) which occupies the least memory. Signed short ranges from -32768 to 32767 and hence we should use unsigned short.

4. Which of the following is a User-defined data type?
a) typedef int Boolean;
b) typedef enum {Mon, Tue, Wed, Thu, Fri} Workdays;
c) struct {char name[10], int age};
d) all of the mentioned

ANSWER:- d
Explanation: typedef and struct are used to define user-defined data types.

5. What is the size of an int data type?
a) 4 Bytes
b) 8 Bytes
c) Depends on the system/compiler
d) Cannot be determined

ANSWER:- c
Explanation: The size of the data types depend on the system.

6. What will be the output of the following C code?


```
#include  <stdio.h>
int main()
{
   signed char chr;
   chr = 128;
   printf("%d\n", chr);
   return 0;
}
```

a) 128
b) -128
c) Depends on the compiler
d) None of the mentioned

ANSWER:- b
Explanation: The range of signed character is from -128 to +127. Since we are assigning a value of 128 to the variable 'chr', the result will be negative. 128 in binary is represented as "1000 0000" for character datatype. As you can see that the sign bit is set to 1, followed by 7 zeros (0), its final decimal value will be -128 (negative 128).

Output:
$ cc pgm2.c
$ a.out
-128

7. What will be the output of the following C code?

```
#include  <stdio.h>
int main()
{
    char c;
    int i = 0;
    FILE *file;
    file = fopen("test.txt", "w+");
    fprintf(file, "%c", 'a');
    fprintf(file, "%c", -1);
     fprintf(file, "%c", 'b');
     fclose(file);
     file = fopen("test.txt", "r");
     while ((c = fgetc(file)) !=  -1)
         printf("%c", c);
     return 0;
 }
```

a) a
b) Infinite loop
c) Depends on what fgetc returns
d) Depends on the compiler

ANSWER:- a
Explanation: None.
Output:
$ cc pgm3.c
$ a.out
a

8. What is short int in C programming?
a) The basic data type of C
b) Qualifier
c) Short is the qualifier and int is the basic data type
d) All of the mentioned

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    float f1 = 0.1;
    if (f1 == 0.1)
        printf("equal\n");
    else
        printf("not equal\n");
}
```

a) equal
b) not equal
c) output depends on the compiler
d) error

ANSWER:- b
Explanation: 0.1 by default is of type double which has different representation than float resulting in inequality even after conversion.
Output:
$ cc pgm4.c
$ a.out
not equal

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    float f1 = 0.1;
    if (f1 == 0.1f)
        printf("equal\n");
    else
        printf("not equal\n");
}
```

a) equal
b) not equal
c) output depends on compiler
d) error

ANSWER:- a
Explanation: 0.1f results in 0.1 to be stored in floating point representations.
Output:
$ cc pgm5.c
$ a.out
equal

3. What will be the output of the following C code on a 32-bit machine?

```c
#include <stdio.h>
int main()
{
    int x = 10000;
    double y = 56;
    int *p = &x;
    double *q = &y;
    printf("p and q are %d and %d", sizeof(p), sizeof(q));
    return 0;
}
```

a) p and q are 4 and 4
b) p and q are 4 and 8
c) compiler error
d) p and q are 2 and 8

ANSWER:- a
Explanation: Size of any type of pointer is 4 on a 32-bit machine.
Output:
$ cc pgm6.c
$ a.out
p and q are 4 and 4

4. Which is correct with respect to the size of the data types?
a) char > int > float
b) int > char > float
c) char < int < double
d) double > char > int

ANSWER:- c
Explanation: char has less bytes than int and int has less bytes than double in any system

5. What will be the output of the following C code on a 64 bit machine?

```c
#include <stdio.h>
union Sti
{
    int nu;
    char m;
};
int main()
{
    union Sti s;
     printf("%d", sizeof(s));
     return 0;
}
```

a) 8
b) 5
c) 9
d) 4

ANSWER:- d
Explanation: Since the size of a union is the size of its maximum data type, here int is the largest data type. Hence the size of the union is 4.
Output:
$ cc pgm7.c
$ a.out
4

6. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    float x = 'a';
    printf("%f", x);
    return 0;
}
```

a) a
b) run time error
c) a.0000000
d) 97.000000

ANSWER:- d
Explanation: Since the ASCII value of a is 97, the same is assigned to the float variable and printed.
Output:
$ cc pgm8.c
$ a.out
97.000000

7. Which of the data types has the size that is variable?
a) int
b) struct
c) float
d) double

ANSWER:- b
Explanation: Since the size of the structure depends on its fields, it has a variable size.

1. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
```

```
4.          enum {ORANGE = 5, MANGO,
    BANANA = 4, PEACH};
5.          printf("PEACH = %d\n",
    PEACH);
6.      }
```

a) PEACH = 3
b) PEACH = 4
c) PEACH = 5
d) PEACH = 6

ANSWER:- c
Explanation: In enum, the value of constant is defined to the recent assignment from left.
Output:
$ cc pgm1.c
$ a.out
PEACH = 5

2. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          printf("C
    programming %s", "Class by\n%s
    Sanfoundry", "WOW");
5.      }
```

a)

C programming Class by

WOW Sanfoundry

b) C programming Class by\n%s Sanfoundry
c)

C programming Class by

%s Sanfoundry

d) Compilation error

ANSWER:- c
Explanation: This program has only one %s within first double quotes, so it does not read the string "WOW".
The %s along with the Sanfoundry is not read as a format modifier while new line character prints the new line.
Output:
$ cc pgm2.c
$ a.out
C programming Class by
%s Sanfoundry

3. In the following code snippet, character pointer str holds a reference to the string
_____

```
char *str =  "Sanfoundry.com\0" "training
classes";
```

a) Sanfoundry.com
b) Sanfoundry.com\0training classes
c) Sanfoundry.comtraining classes
d) Invalid declaration

ANSWER:- b
Explanation: '\0' is accepted as a char in the string. Even though strlen will give length of string "Sanfoundry.com", in memory str is pointing to entire string including training classes.

4. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #define a 10
3.      int main()
4.      {
5.          const int a = 5;
6.          printf("a = %d\n", a);
7.      }
```

a) a = 5
b) a = 10
c) Compilation error
d) Runtime error

ANSWER:- c
Explanation: The #define substitutes a with 10 without leaving any identifier, which results in Compilation error.
Output:
$ cc pgm3.c
pgm3.c: In function 'main':
pgm3.c:5: error: expected identifier or '(' before numeric constant

5. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int var = 010;
```

```
5.          printf("%d", var);
6.      }
```

a) 2
b) 8
c) 9
d) 10

ANSWER:- b
Explanation: 010 is octal representation of 8.
Output:
$ cc pgm4.c
$ a.out
8

6. What will be the output of the following C function?

```
1.      #include <stdio.h>
2.      enum birds {SPARROW, PEACOCK,
    PARROT};
3.      enum animals {TIGER = 8,
    LION, RABBIT, ZEBRA};
4.      int main()
5.      {
6.          enum birds m = TIGER;
7.          int k;
8.          k = m;
9.          printf("%d\n", k);
10.          return 0;
11.      }
```

a) 0
b) Compile time error
c) 1
d) 8

ANSWER:- d
Explanation: m is an integer constant, hence it is compatible.
Output:
$ cc pgm5.c
$ a.out
8

7. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #define MAX 2
3.      enum bird {SPARROW = MAX + 1,
    PARROT = SPARROW + MAX};
4.      int main()
5.      {
6.          enum bird b = PARROT;
7.          printf("%d\n", b);
8.          return 0;
9.      }
```

a) Compilation error
b) 5
c) Undefined value
d) 2

ANSWER:- b
Explanation: MAX value is 2 and hence PARROT will have value 3 + 2.
Output:
$ cc pgm6.c
$ a.out
5

8. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #include <string.h>
3.      int main()
4.      {
5.          char *str = "x";
6.          char c = 'x';
7.          char ary[1];
8.          ary[0] = c;
9.          printf("%d %d",
    strlen(str), strlen(ary));
10.          return 0;
11.      }
```

a) 1 1
b) 2 1
c) 2 2
d) 1 (undefined value)

ANSWER:- d
Explanation: str is null terminated, but ary is not null terminated.
Output:
$ cc pgm7.c
$ a.out
1 5

1. enum types are processed by _____
a) Compiler
b) Preprocessor
c) Linker
d) Assembler

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
printf("sanfoundry\rclass\n");
        return 0;
}
```

a) sanfoundryclass
b)

sanfoundry

class

c) classundry
d) sanfoundry

ANSWER:- c
Explanation: r is carriage return and moves the cursor back. sanfo is replaced by class.
Output:
$ cc pgm8.c
$ a.out
classundry

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
printf("sanfoundry\r\nclass\n");
        return 0;
}
```

a) sanfoundryclass
b)

sanfoundry

class

c) classundry
d) sanfoundry

ANSWER:- b
Explanation: rn combination makes the cursor move to the next line.
Output:
$ cc pgm9.c
$ a.out
sanfoundry
class

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    const int p;
    p = 4;
    printf("p is %d", p);
    return 0;
}
```

a) p is 4
b) Compile time error
c) Run time error
d) p is followed by a garbage value

ANSWER:- b
Explanation: Since the constant variable has to be declared and defined at the same time, not doing it results in an error.
Output:
$ cc pgm10.c
pgm10.c: In function ‘main’:
pgm10.c:5: error: assignment of read-only variable ‘p’

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 4;
    int *const p = &k;
    int r = 3;
    p = &r;
    printf("%d", p);
}
```

a) Address of k
b) Address of r
c) Compile time error
d) Address of k + address of r

ANSWER:- c
Explanation: Since the pointer p is declared to be constant, trying to assign it with a new value results in an error.
Output:
$ cc pgm11.c
pgm11.c: In function ‘main’:
pgm11.c:7: error: assignment of read-only variable ‘p’
pgm11.c:8: warning: format ‘%d’ expects type ‘int’, but argument 2 has type ‘int * const’

6. Which of the following statement is false?
a) Constant variables need not be defined as they are declared and can be defined later
b) Global constant variables are initialized to zero
c) const keyword is used to define constant values
d) You cannot reassign a value to a constant variable

ANSWER:- a
Explanation: Since the constant variable has to be declared and defined at the same time, not doing it results in an error.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int const k = 5;
5.      k++;
6.      printf("k is %d", k);
7.  }
```

a) k is 6
b) Error due to const succeeding int
c) Error, because a constant variable can be changed only twice
d) Error, because a constant variable cannot be changed

ANSWER:- d
Explanation: Constant variable has to be declared and defined at the same time. Trying to change it results in an error.
Output:
$ cc pgm12.c
pgm12.c: In function ‘main’:
pgm12.c:5: error: increment of read-only variable ‘k’

8. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int const print()
3.  {
4.      printf("Sanfoundry.com");
5.      return 0;
6.  }
7.  void main()
8.  {
9.      print();
10.  }
```

a) Error because function name cannot be preceded by const
b) Sanfoundry.com
c) Sanfoundry.com is printed infinite times
d) Blank screen, no output

ANSWER:- b
Explanation: None.
Output:
$ cc pgm13.c
$ a.out

1. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void foo(const int *);
3.  int main()
4.  {
5.      const int i = 10;
6.      printf("%d ", i);
7.      foo(&i);
8.      printf("%d", i);
9.
10.  }
11.  void foo(const int *i)
12.  {
13.      *i = 20;
14.  }
```

a) Compile time error
b) 10 20
c) Undefined value
d) 10

ANSWER:- a
Explanation: Cannot change a const type value.
Output:
$ cc pgm1.c
pgm1.c: In function ‘foo’:

pgm1.c:13: error: assignment of read-only location ‘*i’

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    const int i = 10;
    int *ptr = &i;
    *ptr = 20;
    printf("%d\n", i);
    return 0;
}
```

a) Compile time error
b) Compile time warning and printf displays 20
c) Undefined behaviour
d) 10

ANSWER:- b
Explanation: Changing const variable through non-constant pointers invokes compiler warning.
Output:
$ cc pgm2.c
pgm2.c: In function ‘main’:
pgm2.c:5: warning: initialization discards qualifiers from pointer target type
$ a.out
20

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    j = 10;
    printf("%d\n", j++);
    return 0;
}
```

a) 10
b) 11
c) Compile time error
d) 0

ANSWER:- c
Explanation: Variable j is not defined.
Output:
$ cc pgm3.c
pgm3.c: In function ‘main’:
pgm3.c:4: error: ‘j’ undeclared (first use in this function)
pgm3.c:4: error: (Each undeclared identifier is reported only once
pgm3.c:4: error: for each function it appears in.)

4. Will the following C code compile without any error?

```
#include <stdio.h>
int main()
{
    for (int k = 0; k < 10; k++);
        return 0;
}
```

a) Yes
b) No
c) Depends on the C standard implemented by compilers
d) Error

ANSWER:- c
Explanation: Compilers implementing C90 do not allow this, but compilers implementing C99 allow it.
Output:
$ cc pgm4.c
pgm4.c: In function ‘main’:
pgm4.c:4: error: ‘for’ loop initial declarations are only allowed in C99 mode
pgm4.c:4: note: use option -std=c99 or -std=gnu99 to compile your code

5. Will the following C code compile without any error?

```
#include <stdio.h>
int main()
{
    int k;
    {
        int k;
        for (k = 0; k < 10; k++);
    }
}
```

a) Yes
b) No
c) Depends on the compiler
d) Depends on the C standard implemented by compilers

ANSWER:- a
Explanation: There can be blocks inside the block. But within a block, variables have only block scope.
Output:
$ cc pgm5.c

6. Which of the following declaration is not supported by C?
a) String str;
b) char *str;
c) float str = 3e2;
d) Both “String str;” and “float str = 3e2;”

ANSWER:- a
Explanation: It is legal in Java, but not in C.

7. Which of the following format identifier can never be used for the variable var?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *var = "Advanced
    Training in C by
    Sanfoundry.com";
5.    }
```

a) %f
b) %d
c) %c
d) %s

ANSWER:- a
Explanation: %c can be used to print the indexed position.
%d can still be used to display its ASCII value.
%s is recommended.
%f cannot be used for the variable var

1. Which of the following declaration is illegal?
a) char *str = “Best C programming classes by Sanfoundry”;
b) char str[] = “Best C programming classes by Sanfoundry”;
c) char str[20] = “Best C programming classes by Sanfoundry”;
d) char[] str = “Best C programming classes by Sanfoundry”;

ANSWER:- d
Explanation: char[] str is a declaration in Java, but not in C.

2. Which keyword is used to prevent any changes in the variable within a C program?
a) immutable
b) mutable
c) const
d) volatile

ANSWER:- c
Explanation: const is a keyword constant in C program.

3. Which of the following is not a pointer declaration?
a) char a[10];
b) char a[] = {‘1’, ‘2’, ‘3’, ‘4’};
c) char *str;
d) char a;

ANSWER:- d
Explanation: Array declarations are pointer declarations.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int k = 4;
5.        float k = 4;
6.        printf("%d", k)
7.    }
```

a) Compile time error
b) 4
c) 4.0000000
d) 4.4

ANSWER:- a
Explanation: Since the variable k is defined both as integer and as float, it results in an error.
Output:
$ cc pgm8.c
pgm8.c: In function ‘main’:
pgm8.c:5: error: conflicting types for ‘k’
pgm8.c:4: note: previous definition of ‘k’ was here
pgm8.c:6: warning: format ‘%d’ expects type ‘int’, but argument 2 has type ‘double’
pgm8.c:7: error: expected ‘;’ before ‘}’ token

5. Which of the following statement is false?
a) A variable defined once can be defined again with different scope
b) A single variable cannot be defined with two different types in the same scope
c) A variable must be declared and defined at the same time
d) A variable refers to a location in memory

ANSWER:- c
Explanation: It is not an error if the variable is declared and not defined. For example – extern declarations.

6. A variable declared in a function can be used in main().
a) True
b) False
c) True if it is declared static
d) None of the mentioned

ANSWER:- b
Explanation: Since the scope of the variable declared within a function is restricted only within that function, so the above statement is false.

7. The name of the variable used in one function cannot be used in another function.
a) True
b) False

ANSWER:- b
Explanation: Since the scope of the variable declared within a function is restricted only within that function, the same name can be used to declare another variable in another function.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = -3;
5.        int k = i % 2;
6.        printf("%d\n", k);
7.    }
```

a) Compile time error
b) -1
c) 1
d) Implementation defined

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 3;
5.        int l = i / -2;
6.        int k = i % -2;
7.        printf("%d %d\n", l, k);
8.        return 0;
9.    }
```

a) Compile time error
b) -1 1
c) 1 -1
d) Implementation defined

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 5;
5.        i = i / 3;
6.        printf("%d\n", i);
7.        return 0;
8.    }
```

a) Implementation defined
b) 1
c) 3
d) Compile time error

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
1.   #include <stdio.h>
2.    int main()
3.    {
4.        int i = -5;
5.        i = i / 3;
6.        printf("%d\n", i);
7.        return 0;
8.    }
```

a) Implementation defined
b) -1
c) -3
d) Compile time error

ANSWER:- b
Explanation: None.

5. What will be the final value of x in the following C code?

```c
#include <stdio.h>
void main()
{
    int x = 5 * 9 / 3 + 9;
}
```

a) 3.75
b) Depends on compiler
c) 24
d) 3

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```c
#include <stdio.h>
void main()
{
    int x = 5.3 % 2;
    printf("Value of x
is %d", x);
}
```

a) Value of x is 2.3
b) Value of x is 1
c) Value of x is 0.3
d) Compile time error

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```c
#include <stdio.h>
void main()
{
    int y = 3;
    int x = 5 % 2 * 3 / 2;
    printf("Value of x
is %d", x);
}
```

a) Value of x is 1
b) Value of x is 2
c) Value of x is 3
d) Compile time error

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```c
#include <stdio.h>
void main()
{
    int a = 3;
    int b = ++a + a++ + --a;
    printf("Value of b
is %d", b);
}
```

a) Value of x is 12
b) Value of x is 13
c) Value of x is 10
d) Undefined behaviour

ANSWER:- d
Explanation: None.

2. What is the precedence of arithmetic operators (from highest to lowest)?
a) %, *, /, +, –
b) %, +, /, *, –
c) +, -, %, *, /
d) %, +, -, *, /

ANSWER:- a
Explanation: None.

3. Which of the following is not an arithmetic operation?
a) a * = 10;
b) a / = 10;
c) a ! = 10;
d) a % = 10;

ANSWER:- c
Explanation: None.

4. Which of the following data type will throw an error on modulus operation(%)?
a) char
b) short
c) int
d) float

ANSWER:- d
Explanation: None.

5. Which among the following are the fundamental arithmetic operators, i.e, performing the desired operation can be done using that operator only?
a) +, –
b) +, -, %
c) +, -, *, /
d) +, -, *, /, %

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10;
    double b = 5.6;
    int c;
    c = a + b;
    printf("%d", c);
}
```

a) 15
b) 16
c) 15.6
d) 10

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10, b = 5, c = 5;
    int d;
    d = a == (b + c);
    printf("%d", d);
}
```

a) Syntax error
b) 1
c) 10
d) 5

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 1, y = 0, z = 5;
    int a = x && y || z++;
    printf("%d", z);
}
```

a) 6
b) 5
c) 0
d) Varies

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 1, y = 0, z = 5;
    int a = x && y && z++;
    printf("%d", z);
}
```

a) 6
b) 5
c) 0
d) Varies

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1, y = 0, z = 3;
    x > y ? printf("%d", z) :
return z;
}
```

a) 3
b) 1
c) Compile time error
d) Run time error

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 1, z = 3;
    int y = x << 3;
    printf(" %d\n", y);
```

```
}
```

a) -2147483648
b) -1
c) Run time error
d) 8

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 0, y = 2, z = 3;
    int a = x & y | z;
    printf("%d", a);
}
```

a) 3
b) 0
c) 2
d) Run time error

ANSWER:- a
Explanation: None.

6. What will be the final value of j in the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    if (i && (j = i + 10))
        //do something
        ;
}
```

a) 0
b) 10
c) Depends on the compiler
d) Depends on language standard

ANSWER:- a
Explanation: None.

7. What will be the final value of j in the following C code?

```
#include <stdio.h>
int main()
{
    int i = 10, j = 0;
    if (i || (j = i + 10))
        //do something
        ;
}
```

a) 0
b) 20
c) Compile time error
d) Depends on language standard

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 1;
    if (i++ && (i == 1))
        printf("Yes\n");
    else
        printf("No\n");
}
```

a) Yes
b) No
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

1. Are logical operator sequence points?
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

2. Do logical operators in the C language are evaluated with the short circuit?
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

3. What is the result of logical or relational expression in C?
a) True or False
b) 0 or 1
c) 0 if an expression is false and any positive number if an expression is true
d) None of the mentioned

ANSWER:- b
Explanation: None.

4. What will be the final value of d in the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10, b = 5, c = 5;
    int d;
    d = b + c == a;
    printf("%d", d);
}
```

a) Syntax error
b) 1
c) 5
d) 10

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10, b = 5, c = 3;
    b != !a;
    c = !!a;
    printf("%d\t%d", b, c);
}
```

a) 5 1
b) 0 3
c) 5 3
d) 1 1

ANSWER:- a
Explanation: None.

6. Which among the following is NOT a logical or relational operator?
a) !=
b) ==
c) | |
d) =

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10;
    if (a == a--)
        printf("TRUE 1\t");
    a = 10;
    if (a == --a)
        printf("TRUE 2\t");
 }
```

a) TRUE 1
b) TRUE 2
c) TRUE 1 TRUE 2
d) Compiler Dependent

ANSWER:- d
Explanation: This is a sequence point problem and hence the result will be implementation dependent.

8. Relational operators cannot be used on _____
a) structure
b) long
c) strings
d) float

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    float x = 0.1;
    if (x == 0.1)
        printf("Sanfoundry");
    else
        printf("Advanced C Classes");
}
```

a) Advanced C Classes
b) Sanfoundry
c) Run time error
d) Compile time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    float x = 0.1;
    printf("%d, ", x);
    printf("%f", x);
}
```

a) 0.100000, junk value
b) Junk value, 0.100000
c) 0, 0.100000
d) 0, 0.999999

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?
(Initial values: x= 7, y = 8)

```
#include <stdio.h>
void main()
{
    float x;
    int y;
    printf("enter two
numbers \n");
    scanf("%f %f", &x, &y);
    printf("%f, %d", x, y);
}
```

a) 7.000000, 7
b) Run time error
c) 7.000000, junk
d) Varies

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    double x = 123828749.66;
    int y = x;
    printf("%d\n", y);
    printf("%lf\n", y);
}
```

a) 0, 0.0
b) 123828749, 123828749.66
c) 12382874, 12382874.0
d) 123828749, 0.000000

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 97;
    char y = x;
    printf("%c\n", y);
}
```

a) a
b) b
c) 97
d) Run time error

ANSWER:- a
Explanation: None.

6. When double is converted to float, then the value is?
a) Truncated
b) Rounded
c) Depends on the compiler
d) Depends on the standard

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    unsigned int i = 23;
    signed char c = -23;
    if (i > c)
        printf("Yes\n");
    else if (i < c)
```

```
9.              printf("No\n");
10.     }
```

a) Yes
b) No
c) Depends on the compiler
d) Depends on the operating system

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 23;
5.        char c = -23;
6.        if (i < c)
7.            printf("Yes\n");
8.        else
9.            printf("No\n");
10.    }
```

a) Yes
b) No
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

1. function tolower(c) defined in library <ctype.h> works for _____
a) Ascii character set
b) Unicode character set
c) Ascii and utf-8 but not EBCDIC character set
d) Any character set

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code considering the size of a short int is 2, char is 1 and int is 4 bytes?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        short int i = 20;
5.        char c = 97;
6.        printf("%d, %d, %d\n",
      sizeof(i), sizeof(c), sizeof(c
      + i));
7.        return 0;
8.    }
```

a) 2, 1, 2
b) 2, 1, 1
c) 2, 1, 4
d) 2, 2, 8

ANSWER:- c
Explanation: None.

Check this: C Books | BCA MCQs

3. Which type of conversion is NOT accepted?
a) From char to int
b) From float to char pointer
c) From negative int to char
d) From double to char

ANSWER:- b
Explanation: Conversion of a float to pointer type is not allowed.

4. What will be the data type of the result of the following operation?

```
(float)a * (int)b / (long)c * (double)d
```

a) int
b) long
c) float
d) double

ANSWER:- d
Explanation: None.

5. Which of the following type-casting have chances for wrap around?
a) From int to float
b) From int to char
c) From char to short
d) From char to int

ANSWER:- b
Explanation: None.

6. Which of the following typecasting is accepted by C?
a) Widening conversions
b) Narrowing conversions
c) Widening & Narrowing conversions
d) None of the mentioned

ANSWER:- c
Explanation: None.

7. When do you need to use type-conversions?
a) The value to be stored is beyond the max limit
b) The value to be stored is in a form not supported by that data type
c) To reduce the memory in use, relevant to the value
d) All of the mentioned

ANSWER:- d
Explanation: None.

1. What is the difference between the following 2 codes?

```
#include <stdio.h> //Program 1
int main()
{
    int d, a = 1, b = 2;
    d =  a++ + ++b;
    printf("%d %d %d", d, a, b);
}
```

```
#include <stdio.h> //Program 2
int main()
{
    int d, a = 1, b = 2;
    d =  a++ +++b;
    printf("%d %d %d", d, a, b);
}
```

a) No difference as space doesn't make any difference, values of a, b, d are same in both the case
b) Space does make a difference, values of a, b, d are different
c) Program 1 has syntax error, program 2 is not
d) Program 2 has syntax error, program 1 is not

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 1, b = 1, c;
    c = a++ + b;
    printf("%d, %d", a, b);
}
```

a) a = 1, b = 1
b) a = 2, b = 1
c) a = 1, b = 2
d) a = 2, b = 2

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 1, b = 1, d = 1;
    printf("%d, %d, %d", ++a + ++a+a++, a++ + ++b, ++d + d++ + a++);
}
```

a) 15, 4, 5
b) 9, 6, 9
c) 9, 3, 5
d) Undefined (Compiler Dependent)

ANSWER:- d
Explanation: None.

4. For which of the following, "PI++;" code will fail?
a) #define PI 3.14
b) char *PI = "A";
c) float PI = 3.14;
d) none of the Mentioned

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 10, b = 10;
    if (a = 5)
    b--;
    printf("%d, %d", a, b--);
}
```

a) a = 10, b = 9
b) a = 10, b = 8
c) a = 5, b = 9
d) a = 5, b = 8

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    int j = i++ + i;
    printf("%d\n", j);
}
```

a) 0
b) 1
c) 2
d) Compile time error

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
 #include <stdio.h>
int main()
{
    int i = 2;
    int j = ++i + i;
    printf("%d\n", j);
}
```

a) 6
b) 5
c) 4
d) Compile time error

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>

int main()
{
    int i = 2;
    int i = i++ + i;
    printf("%d\n", i);
}
```

a) = operator is not a sequence point
b) ++ operator may return value with or without side effects
c) it can be evaluated as (i++)+i or i+(++i)
d) = operator is a sequence point

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int c = 2 ^ 3;
    printf("%d\n", c);
}
```

a) 1
b) 8
c) 9
d) 0

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    unsigned int a = 10;
    a = ~a;
    printf("%d\n", a);
}
```

a) -9
b) -10
c) -11
d) 10

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    if (7 & 8)
```

```
5.          printf("Honesty");
6.              if ((~7 & 0x000f) ==
    8)
7.                  printf("is the
    best policy\n");
8.      }
```

a) Honesty is the best policy
b) Honesty
c) is the best policy
d) No output

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int a = 2;
5.      if (a >> 1)
6.         printf("%d\n", a);
7.  }
```

a) 0
b) 1
c) 2
d) No Output

ANSWER:- c
Explanation: None.

5. Comment on the output of the following C code.

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int i, n, a = 4;
5.      scanf("%d", &n);
6.      for (i = 0; i < n; i++)
7.          a = a * 2;
8.  }
```

a) Logical Shift left
b) No output
c) Arithmetic Shift right
d) Bitwise exclusive OR

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int x = 97;
5.      int y = sizeof(x++);
6.      printf("x is %d", x);
7.  }
```

a) x is 97
b) x is 98
c) x is 99
d) Run time error

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int x = 4, y, z;
5.      y = --x;
6.      z = x--;
7.      printf("%d%d%d", x, y,
    z);
8.  }
```

a) 3 2 3
b) 2 2 3
c) 3 2 2
d) 2 3 3

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int x = 4;
5.      int *p = &x;
6.      int *k = p++;
7.      int r = p - k;
8.      printf("%d", r);
9.  }
```

a) 4
b) 8
c) 1
d) Run time error

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a = 5, b = -7, c = 0, d;
    d = ++a && ++b || ++c;
    printf("\n%d%d%d%d", a, b, c, d);
}
```

a) 6 -6 0 0
b) 6 -5 0 1
c) -6 -6 0 1
d) 6 -6 0 1

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a = -5;
    int k = (a++, ++a);
    printf("%d\n", k);
}
```

a) -3
b) -5
c) 4
d) Undefined

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2;
    x = x << 1;
    printf("%d\n", x);
}
```

a) 4
b) 1
c) Depends on the compiler
d) Depends on the endianness of the machine

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = -2;
    x = x >> 1;
    printf("%d\n", x);
}
```

a) 1
b) -1
c) 2^{31} – 1 considering int to be 4 bytes
d) Either -1 or 1

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    if (~0 == 1)
        printf("yes\n");
    else
        printf("no\n");
}
```

a) yes
b) no
c) compile time error
d) undefined

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
```

```
3.    {
4.        int x = -2;
5.        if (!0 == 1)
6.            printf("yes\n");
7.        else
8.            printf("no\n");
9.    }
```

a) yes
b) no
c) run time error
d) undefined

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int y = 0;
5.        if (1 |(y = 1))
6.            printf("y is %d\n",
    y);
7.        else
8.            printf("%d\n", y);
9.
10.    }
```

a) y is 1
b) 1
c) run time error
d) undefined

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int y = 1;
5.        if (y & (y = 2))
6.            printf("true %d\n",
    y);
7.        else
8.            printf("false %d\n",
    y);
9.
10.    }
```

a) true 2
b) false 2
c) either true 2 or false 2
d) true 1

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int x = 0;
5.        if (x = 0)
6.            printf("Its zero\n");
7.        else
8.            printf("Its not
    zero\n");
9.    }
```

a) Its not zero
b) Its zero
c) Run time error
d) None

ANSWER:- a
Explanation: In the above C code, we assign a zero value to the variable x. In the if condition, we are reassigning a value of zero to x. Remember, we are “NOT” comparing its values to zero (you can see that it is a single ‘=’ sign, it’s not a double ‘==’ sign). Hence, the if-condition becomes false and the printf() function of the else condition will be executed which will display “Its not zero”.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int k = 8;
5.        int x = 0 == 1 && k++;
6.        printf("%d%d\n", x, k);
7.    }
```

a) 0 9
b) 0 8
c) 1 8

d) 1 9

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char a = 'a';
    int x = (a % 10)++;
    printf("%d\n", x);
}
```

a) 6
b) Junk value
c) Compile time error
d) 7

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code snippet?

```
#include <stdio.h>
void main()
{
    1 < 2 ? return 1: return 2;
}
```

a) returns 1
b) returns 2
c) Varies
d) Compile time error

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code snippet?

```
#include <stdio.h>
void main()
{
    unsigned int x = -5;
    printf("%d", x);
}
```

a) Run time error
b) Aries
c) -5
d) 5

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 1;
    x *= x + y;
    printf("%d\n", x);
    return 0;
}
```

a) 5
b) 6
c) Undefined behaviour
d) Compile time error

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 2;
    x /= x / y;
    printf("%d\n", x);
    return 0;
}
```

a) 2
b) 1
c) 0.5
d) Undefined behaviour

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1, y = 0;
    x &&= y;
    printf("%d\n", x);
```

```
7.      }
```

a) Compile time error
b) 1
c) 0
d) Undefined behaviour

ANSWER:- a
Explanation: None.

1. What is the type of the following assignment expression if x is of type float and y is of type int?

$y = x + y;$

a) int
b) float
c) there is no type for an assignment expression
d) double

ANSWER:- a
Explanation: None.

2. What will be the value of the following C expression?

$(x = foo()) != 1$ considering $foo()$ returns 2

a) 2
b) True
c) 1
d) 0

ANSWER:- c
Explanation: The C language sub-expression "x = foo()" will assign a value of 2 to the variable 'x'. Then, it will check if this value is not equal to 1 which is true and hence the result will be 1.

3. Operation "a = a * b + a" can also be written as _____

a) a *= b + 1;
b) (c = a * b)!=(a = c + a);
c) a = (b + 1)* a;
d) All of the mentioned

ANSWER:- d
Explanation: None.

4. What will be the final value of c in the following C statement? (Initial value: c = 2)

```
1. c <<= 1;
```

a) c = 1;
b) c = 2;
c) c = 3;
d) c = 4;

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int a = 1, b = 2;
5.        a += b -= a;
6.        printf("%d %d", a, b);
7.    }
```

a) 1 1
b) 1 2
c) 2 1
d) 2 2

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int a = 4, n, i, result
      = 0;
5.        scanf("%d", n);
6.        for (i = 0;i < n; i++)
7.        result += a;
8.    }
```

a) Addition of a and n
b) Subtraction of a and n
c) Multiplication of a and n
d) Division of a and n

ANSWER:- c
Explanation: None.

7. Which of the following is an invalid assignment operator?
a) a %= 10;
b) a /= 10;
c) a |= 10;
d) None of the mentioned

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 0;
    int z = (y++) ? y == 1 && x : 0;
    printf("%d\n", z);
    return 0;
}
```

a) 0
b) 1
c) Undefined behaviour
d) Compile time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1;
    int y =  x == 1 ? getchar(): 2;
    printf("%d\n", y);
}
```

a) Compile time error
b) Whatever character getchar function returns
c) Ascii value of character getchar function returns
d) 2

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1;
    short int i = 2;
    float f = 3;
    if (sizeof((x == 2) ? f : i) == sizeof(float))
        printf("float\n");
    else if (sizeof((x == 2) ? f : i) == sizeof(short int))
        printf("short int\n");
}
```

a) float
b) short int
c) Undefined behaviour
d) Compile time error

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 2;
    int b = 0;
    int y = (b == 0) ? a :(a > b) ? (b = 1): a;
    printf("%d\n", y);
}
```

a) Compile time error
b) 1
c) 2
d) Undefined behaviour

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int y = 1, x = 0;
    int l = (y++, x++) ? y : x;
    printf("%d\n", l);
}
```

a) 1
b) 2
c) Compile time error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 8;
    int m = 7;
    int z = k < m ? k++ :
m++;
    printf("%d", z);
}
```

a) 7
b) 8
c) Run time error
d) 15

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 8;
    int m = 7;
    int z = k < m ? k = m :
m++;
    printf("%d", z);
}
```

a) Run time error
b) 7
c) 8
d) Depends on compiler

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    1 < 2 ? return 1 :
return 2;
}
```

a) returns 1
b) returns 2
c) Varies
d) Compile time error

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 8;
    int m = 7;
    k < m ? k++ : m = k;
    printf("%d", k);
}
```

a) 7
b) 8
c) Compile time error
d) Run time error

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 8;
    int m = 7;
    k < m ? k = k + 1 : m =
m + 1;
    printf("%d", k);
}
```

a) Compile time error
b) 9
c) 8
d) Run time error

ANSWER:- a
Explanation: None.

3. What will be the final values of a and c in the following C statement? (Initial values: a = 2, c = 1)

```
c = (c) ? a = 0 : 2;
```

a) a = 0, c = 0;
b) a = 2, c = 2;
c) a = 2, c = 2;
d) a = 1, c = 2;

ANSWER:- a
Explanation: None.

4. What will be the data type of the following expression? (Initial data type: a = int, var1 = double, var2 = float)

```
expression (a < 50)? var1 : var2;
```

a) int
b) float
c) double
d) Cannot be determined

ANSWER:- c
Explanation: None.

5. Which expression has to be present in the following?

exp1 ? exp2 : exp3;

a) exp1
b) exp2
c) exp3
d) all of the mentioned

ANSWER:- d
Explanation: None.

6. What will be the final value of c in the following C code snippet? (Initial values: a = 1, b = 2, c = 1)

```
      c += (-c) ? a : b;
```

a) Syntax Error
b) c = 1
c) c = 2
d) c = 3

ANSWER:- c
Explanation: None.

7. The following C code can be rewritten as _____

```
c = (n) ? a : b;
```

a)

```
   if (!n)c = b;
   else c = a;
```

b)

```
   if (n <;= 0)c = b;
   else c = a;
```

c)

```
   if (n > 0)c = a;
   else c = b;
```

d) All of the mentioned

ANSWER:- a
Explanation: None.

1. What will be the output of the following C function?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        reverse(1);
5.    }
6.    void reverse(int i)
7.    {
8.        if (i > 5)
9.            exit(0);
10.       printf("%d\n", i);
11.       return reverse(i++);
12.    }
```

a) 1 2 3 4 5
b) 1 2 3 4
c) Compile time error
d) Stack overflow

ANSWER:- d
Explanation: None.

2. What will be the output of the following C function?

```
1.    #include <stdio.h>
2.    void reverse(int i);
3.    int main()
4.    {
5.        reverse(1);
6.    }
```

```
7.     void reverse(int i)
8.     {
9.         if (i > 5)
10.             return ;
11.         printf("%d ", i);
12.         return reverse((i++,
    i));
13.     }
```

a) 1 2 3 4 5
b) Segmentation fault
c) Compilation error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

3. In expression i = g() + f(), first function called depends on _____
a) Compiler
b) Associativiy of () operator
c) Precedence of () and + operator
d) Left to write of the expression

ANSWER:- a
Explanation: None.

4. What will be the final values of i and j in the following C code?

```
1.     #include <stdio.h>
2.     int x = 0;
3.     int f()
4.     {
5.         if (x == 0)
6.             return x + 1;
7.         else
8.             return x - 1;
9.     }
10.     int g()
11.     {
12.         return x++;
13.     }
14.     int main()
15.     {
16.         int i = (f() + g()) ||
    g();
17.         int j = g() || (f() +
    g());
18.     }
```

a) i value is 1 and j value is 1
b) i value is 0 and j value is 0
c) i value is 1 and j value is undefined
d) i and j value are undefined

ANSWER:- d
Explanation: None.

5. What will be the final values of i and j in the following C code?

```
1.     #include <stdio.h>
2.     int x = 0;
3.     int f()
4.     {
5.         if (x == 0)
6.             return x + 1;
7.         else
8.             return x - 1;
9.     }
10.     int g()
11.     {
12.         return x++;
13.     }
14.     int main()
15.     {
16.         int i = (f() + g()) |
    g(); //bitwise or
17.         int j = g() | (f() +
    g()); //bitwise or
18.     }
```

a) i value is 1 and j value is 1
b) i value is 0 and j value is 0
c) i value is 1 and j value is undefined
d) i and j value are undefined

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         int x = 2, y = 0;
5.         int z = y && (y |= 10);
6.         printf("%d\n", z);
7.         return 0;
8.     }
```

a) 1
b) 0
c) Undefined behaviour due to order of evaluation
d) 2

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 0;
    int z = (y++) ? 2 : y == 1 && x;
    printf("%d\n", z);
    return 0;
}
```

a) 0
b) 1
c) 2
d) Undefined behaviour

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 0;
    int z;
    z = (y++, y);
    printf("%d\n", z);
    return 0;
}
```

a) 0
b) 1
c) Undefined behaviour
d) Compilation error

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 0, l;
    int z;
    z = y = 1, l = x && y;
    printf("%d\n", l);
    return 0;
}
```

a) 0
b) 1
c) Undefined behaviour due to order of evaluation can be different
d) Compilation error

ANSWER:- b
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int y = 2;
    int z = y +(y = 10);
    printf("%d\n", z);
}
```

a) 12
b) 20
c) 4
d) Either 12 or 20

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int b = 5 - 4 + 2 * 5;
    printf("%d", b);
}
```

a) 25
b) -5
c) 11
d) 16

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int b = 5 & 4 & 6;
    printf("%d", b);
}
```

a) 5
b) 6
c) 3
d) 4

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int b = 5 & 4 | 6;
    printf("%d", b);
}
```

a) 6
b) 4
c) 1
d) 0

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int b = 5 + 7 * 4 - 9 *
(3, 2);
    printf("%d", b);
}
```

a) 6
b) 15
c) 13
d) 21

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int h = 8;
    int b = (h++, h++);
    printf("%d%d\n", b, h);
}
```

a) 10 10
b) 10 9
c) 9 10
d) 8 10

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int h = 8;
    int b = h++ + h++ + h++;
    printf("%d\n", h);
}
```

a) 9
b) 10
c) 12
d) 11

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int h = 8;
    int b = 4 * 6 + 3 * 4 <
3 ? 4 : 3;
    printf("%d\n", b);
}
```

a) 3
b) 33
c) 34
d) Run time error

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a = 2 + 3 - 4 + 8 - 5 % 4;
    printf("%d\n", a);
}
```

a) 0
b) 8
c) 11
d) 9

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char a = '0';
    char b = 'm';
    int c = a && b || '1';
    printf("%d\n", c);
}
```

a) 0
b) a
c) 1
d) m

ANSWER:- c
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char a = 'A';
    char b = 'B';
    int c = a + b % 3 - 3 * 2;
    printf("%d\n", c);
}
```

a) 65
b) 58
c) 64
d) 59

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 2, y = 2;
    float f = y + x /= x / y;
    printf("%d %f\n", x, f);
    return 0;
}
```

a) 2 4.000000
b) Compile time error
c) 2 3.500000
d) Undefined behaviour

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1, y = 2;
    if (x && y == 1)
        printf("true\n");
    else
        printf("false\n");
}
```

a) true
b) false
c) compile time error
d) undefined behaviour

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1, y = 2;
    int z = x & y == 2;
```

```
6.          printf("%d\n", z);
7.      }
```

a) 0
b) 1
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x = 3, y = 2;
5.      int z = x /= y %= 2;
6.      printf("%d\n", z);
7.  }
```

a) 1
b) Compile time error
c) Floating point exception
d) Segmentation fault

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x = 3, y = 2;
5.      int z = x << 1 > 5;
6.      printf("%d\n", z);
7.  }
```

a) 1
b) 0
c) 3
d) Compile time error

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x = 3; //, y = 2;
5.      const int *p = &x;
6.      *p++;
7.      printf("%d\n", *p);
8.  }
```

a) Increment of read-only location compile error
b) 4
c) Some garbage value
d) Undefined behaviour

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x = 2, y = 2;
5.      int z = x ^ y & 1;
6.      printf("%d\n", z);
7.  }
```

a) 1
b) 2
c) 0
d) 1 or 2

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x = 2, y = 0;
5.      int z = x && y = 1;
6.      printf("%d\n", z);
7.  }
```

a) 0
b) 1
c) Compile time error
d) 2

ANSWER:- c
Explanation: None.

9. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int main()
```

```
{
    int x = 0, y = 2;
    if (!x && y)
        printf("true\n");
    else
        printf("false\n");
}
```

a) True
b) False
c) Compile time error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 0, y = 2;
    int z = ~x & y;
    printf("%d\n", z);
}
```

a) -1
b) 2
c) 0
d) Compile time error

ANSWER:- b
Explanation: None.

1. Which of the following operators has an associativity from Right to Left?
a) <=
b) <<
c) ==
d) +=

ANSWER:- d
Explanation: None.

2. Which operators of the following have same precedence?

a) P and Q
b) Q and R
c) P and R
d) P, Q and R

ANSWER:- b
Explanation: None.

3. Comment on the following statement.

```
n = 1;
printf("%d, %dn", 3*n, n++);
```

a) Output will be 3, 2
b) Output will be 3, 1
c) Output will be 6, 1
d) Output is compiler dependent

ANSWER:- d
Explanation: None.

4. Which of the following option is the correct representation of the following C statement?

```
e = a * b + c / d * f;
```

a) e = (a * (b +(c /(d * f))));
b) e = ((a * b) + (c / (d * f)));
c) e = ((a * b) + ((c / d)* f));
d) Both e = ((a * b) + (c / (d * f))); and e = ((a * b) + ((c / d)* f));

ANSWER:- d
Explanation: Verified by e = 1 * 2 + 3 / 4 * 5; and then using respective braces according to the option.

5. While swapping 2 numbers what precautions to be taken care?

```
b = (b / a);
a = a * b;
b = a / b;
```

a) Data type should be either of short, int and long
b) Data type should be either of float and double
c) All data types are accepted except for (char *)
d) This code doesn't swap 2 numbers

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include<stdio.h>
int main()
{
    int a = 1, b = 2, c = 3,
d = 4, e;
```

```
5.          e = c + d = b * a;
6.          printf("%d, %d\n", e, d);
7.      }
```

a) 7, 4
b) 7, 2
c) 5, 2
d) Syntax error

ANSWER:- d
Explanation: None.

7. Which of the following is the correct order of evaluation for the given expression?

```
a = w % x / y * z;
```

a) % / * =
b) / * % =
c) = % * /
d) * % / =

ANSWER:- a
Explanation: None.

8. Which function in the following expression will be called first?

```
a = func3(6) - func2(4, 5) / func1(1, 2, 3);
```

a) func1();
b) func2();
c) func3();
d) Cannot be predicted

ANSWER:- d
Explanation: None.

9. Which of the following operator has the highest precedence in the following?
a) ()
b) sizeof
c) *
d) +

ANSWER:- a
Explanation: None.

10. Which of the following is a ternary operator?
a) &&
b) >>=
c) ?:
d) ->

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int a = 5 * 3 + 2 - 4;
5.      printf("%d", a);
6.  }
```

a) 13
b) 14
c) 12
d) 1 6

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int a = 2 + 4 + 3 * 5 / 3 - 5;
5.      printf("%d", a);
6.  }
```

a) 7
b) 6
c) 10
d) 9

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  void main()
3.  {
4.      int a = 5 * 3 % 6 - 8 + 3;
5.      printf("%d", a);
6.  }
```

a) 10
b) 2
c) -2
d) -3

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int b = 6;
    int c = 7;
    int a = ++b + c--;
    printf("%d", a);
}
```

a) Run time error
b) 15
c) 13
d) 14

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main(
{
    double b = 8;
    b++;
    printf("%lf", b);
}
```

a) 9.000000
b) 9
c) 9.0
d) Run time error

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    double b = 3 % 0 * 1 - 4 / 2;
    printf("%lf", b);
}
```

a) -2
b) Floating point Exception
c) 1
d) 0

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    double b = 5 % 3 & 4 + 5 * 6;
    printf("%lf", b);
}
```

a) 2
b) 30
c) 2.000000
d) Run time error

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    double b = 3 && 5 & 4 % 3;
    printf("%lf", b);
}
```

a) 3.000000
b) 4.000000
c) 5.000000
d) 1.000000

ANSWER:- d
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    double b = 5 & 3 && 4 || 5 | 6;
    printf("%lf", b);
}
```

a) 1.000000
b) 0.000000
c) 7.000000

d) 2.000000

ANSWER:- a
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 0;
    double b = k++ + ++k + k--;
    printf("%d", k);
}
```

a) 6
b) 1
c) 5
d) undefined

ANSWER:- b
Explanation: None.

1. Which of the following are unary operators?
a) sizeof
b) –
c) ++
d) all of the mentioned

ANSWER:- d
Explanation: None.

2. Where in C the order of precedence of operators do not exist?
a) Within conditional statements, if, else
b) Within while, do-while
c) Within a macro definition
d) None of the mentioned

ANSWER:- d
Explanation: None.

3. Associativity of an operator is _____
a) Right to Left
b) Left to Right
c) Random fashion
d) Both Right to Left and Left to Right

ANSWER:- d
Explanation: None.

4. Which of the following method is accepted for assignment?
a) 5 = a = b = c = d;
b) a = b = c = d = 5;
c) a = b = 5 = c = d;
d) None of the mentioned

ANSWER:- b
Explanation: None.

5. Which of the following is NOT possible with any 2 operators in C?
a) Different precedence, same associativity
b) Different precedence, different associativity
c) Same precedence, different associativity
d) All of the mentioned

ANSWER:- c
Explanation: None.

6. Which of the following is possible with any 2 operators in C?
a) Same associativity, different precedence
b) Same associativity, same precedence
c) Different associativity, different precedence
d) All of the mentioned

ANSWER:- d
Explanation: None.

7. Which of the following operators has the lowest precedence?
a) !=
b) &&
c) ?:
d) ,

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 3, i = 0;
    do {
        x = x++;
        i++;
    } while (i != 3);
    printf("%d\n", x);
 }
```

a) Undefined behaviour
b) Output will be 3
c) Output will be 6
d) Output will be 5

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = -1, b = 4, c = 1, d;
    d = ++a && ++b || ++c;
    printf("%d, %d, %d, %d\n", a, b, c, d);
    return 0;
}
```

a) 0, 4, 2, 1
b) 0, 5, 2, 1
c) -1, 4, 1, 1
d) 0, 5, 1, 0

ANSWER:- a
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int p = 10, q = 20, r;
    if (r = p = 5 || q > 20)
        printf("%d", r);
    else
        printf("No Output\n");
}
```

a) 1
b) 10
c) 20
d) No Output

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 5;
    if (x < 1)
        printf ("hello");
    if (x == 5)
        printf ("hi");
    else
        printf ("no");
 }
```

a) hi
b) hello
c) no
d) error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?


```
#include <stdio.h>
int x;
void main()
{
    if (x)
        printf ("hi");
    else
        printf ("how are u");
}
```

a) hi
b) how are you
c) compile time error
d) error

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 5;
    if (true);
        printf ("hello");
}
```

a) It will display hello
b) It will throw an error
c) Nothing will be displayed
d) Compiler dependent

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 0;
    if (x == 0)
        printf ("hi");
    else
        printf ("how are u");
        printf ("hello");
 }
```

a) hi
b) how are you
c) hello
d) hihello

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 5;
    if (x < 1);
        printf ("Hello");

}
```

a) Nothing
b) Run time error
c) Hello
d) Varies

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
#include <stdio.h>
void main()
{
    double ch;
    printf ("enter a value between 1 to 2:");
    scanf("%lf", &ch);
    switch (ch)
    {
       case 1:
          printf ("1");
          break;
        case 2:
          printf ("2");
          break;
     }
 }
```

a) Compile time error
b) 1
c) 2
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
#include <stdio.h>
void main()
{
    char *ch;
    printf ("enter a value between 1 to 3:");
    scanf("%s", ch);
    switch (ch)
    {
       case "1":
           printf ("1");
           break;
        case "2":
           printf ("2");
           break;
     }
 }
```

a) 1
b) 2
c) Compile time error

d) No Compile time error

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
#include <stdio.h>
void main()
{
    int ch;
    printf ("enter a value between 1 to 2:");
    scanf("%d", &ch);
    switch (ch)
    {
       case 1:
           printf ("1\n");
        default:
           printf ("2\n");
     }
 }
```

a) 1
b) 2
c) 1 2
d) Run time error

ANSWER:- c
Explanation: None.

9. What will be the output of the following C code? (Assuming that we have entered the value 2 in the standard input)

```
#include <stdio.h>
void main()
{
    int ch;
    printf ("enter a value between 1 to 2:");
    scanf("%d", &ch);
    switch (ch)
    {
       case 1:
           printf ("1\n");
           break;
           printf ("Hi");
           default:
              printf ("2\n");
        }
     }
```

a) 1
b) Hi 2
c) Run time error
d) 2

ANSWER:- d
Explanation: None.

10. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
#include <stdio.h>
void main()
{
    int ch;
    printf ("enter a value between 1 to 2:");
    scanf("%d", &ch);
    switch (ch, ch + 1)
    {
       case 1:
           printf ("1\n");
           break;
        case 2:
           printf ("2");
           break;
     }
 }
```

a) 1
b) 2
c) 3
d) Run time error

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 1;
    if (x > 0)
```

```
            printf ("inside if\n");
        else if (x > 0)
            printf ("inside elseif\n");
    }
```

a) inside if
b) inside elseif
c)

inside if

inside elseif

d) compile time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 0;
    if (x++)
        printf ("true\n");
    else if (x == 1)
        printf ("false\n");
}
```

a) true
b) false
c) compile time error
d) undefined behaviour

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 0;
    if (x == 1)
        if (x == 0)
            printf ("inside if\n");
        else
            printf ("inside else if\n");
    else
        printf ("inside else\n");
}
```

a) inside if
b) inside else if
c) inside else
d) compile time error

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 0;
    if (x == 0)
        printf ("true, ");
    else if (x = 10)
        printf ("false, ");
    printf ("%d\n", x);
}
```

a) false, 0
b) true, 0
c) true, 10
d) compile time error

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x = 0;
    if (x == 1)
        if (x >= 0)
            printf ("true\n");
        else
            printf ("false\n");
```

```
10.     }
```

a) true
b) false
c) Depends on the compiler
d) No print statement

ANSWER:- d
Explanation: None.

6. The C statement “”if (a == 1 || b == 2) {}”” can be re-written as _____
a)

```
if (a == 1)
if (b == 2){}
```

b)

```
if (a == 1){}
if (b == 2){}
```

c)

```
if (a == 1){}
else if (b == 2){}
```

d) none of the mentioned

ANSWER:- d
Explanation: None.

7. Which of the following is an invalid if-else statement?
a) if (if (a == 1)){}
b) if (func1 (a)){}
c) if (a){}
d) if ((char) a){}

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int a = 1;
5.        if (a--)
6.            printf ("True");
7.            if (a++)
8.                printf ("False");
9.    }
```

a) True
b) False
c) True False
d) No Output

ANSWER:- a
Explanation: None.

9. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int a = 1;
5.        if (a)
6.            printf ("All is Well
");
7.            printf ("I am
Well\n");
8.        else
9.            printf ("I am not a
River\n");
10.    }
```

a) Output will be All is Well I am Well
b) Output will be I am Well I am not a River
c) Output will be I am Well
d) Compile time errors during compilation

ANSWER:- d
Explanation: None.

10. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        if (printf ("%d", printf
("")))
5.            printf ("We are
Happy");
6.        else if (printf ("1"))
7.            printf ("We are
Sad");
8.    }
```

a) 0We are Happy
b) 1We are Happy
c) 1We are Sad
d) compile time error

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
const int a = 1,  b = 2;
int main()
{
    int x = 1;
    switch (x)
    {
       case a:
          printf ("yes ");
        case b:
          printf ("no\n");
          break;
     }
 }
```

a) yes no
b) yes
c) no
d) Compile time error

ANSWER:- d
Explanation: We are violating a C programming rule which states that in switch-case statements, the labels must be integer constants. When you compile the code, the c compiler will give an error message indicating that the case label is not an integer constant because the labels given in the code are integer variables 'a' and 'b'. You will get the following error message:

```
error: case label does not reduce to an integer constant
    case a:
error: case label does not reduce to an integer constant
    case b:
```

2. What will be the output of the following C code?

```
#include <stdio.h>
#define max(a) a
int main()
{
    int x = 1;
    switch (x)
    {
        case max(2):
           printf ("yes\n");
         case max(1):
           printf ("no\n");
           break;
     }
 }
```

a) yes no
b) yes
c) no
d) Compile time error

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    switch (printf ("Do"))
    {
       case 1:
          printf ("First\n");
          break;
       case 2:
           printf ("Second\n");
          break;
        default:
          printf ("Default\n");
          break;
     }
 }
```

a) Do
b) DoFirst
c) DoSecond
d) DoDefault

ANSWER:- c
Explanation: None.

4. Comment on the output of the following C code.

```
#include <stdio.h>
int main()
{
```

```
4.        int a = 1;
5.        switch (a)
6.        case 1:
7.            printf ("%d", a);
8.        case 2:
9.            printf ("%d", a);
10.        case 3:
11.            printf ("%d", a);
12.        default:
13.            printf ("%d", a);
14.    }
```

a) No error, output is 1111
b) No error, output is 1
c) Compile time error, no break statements
d) Compile time error, case label outside switch statement

ANSWER:- d
Explanation: None.

5. Which datatype can accept the switch statement?
a) int
b) char
c) long
d) all of the mentioned

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int a = 1;
5.        switch (a)
6.        {
7.           case a:
8.              printf ("Case A ");
9.           default:
10.              printf
   ("Default");
11.        }
12.    }
```

a) Output: Case A
b) Output: Default
c) Output: Case A Default
d) Compile time error

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    switch (ch)
3.    {
4.       case 'a':
5.       case 'A':
6.          printf ("true");
7.    }
```

a) if (ch == ‘a’ && ch == ‘A’) printf (“true”);
b)

if (ch == 'a')

if (ch == 'a') printf ("true");

c) if (ch == ‘a’ || ch == ‘A’) printf (“true”);
d) none of the mentioned

ANSWER:- c
Explanation: None

1. The C code ‘for(;;)’ represents an infinite loop. It can be terminated by _____
a) break
b) exit(0)
c) abort()
d) terminate

ANSWER:- a
Explanation: None.

2. What will be the correct syntax for running two variable for loop simultaneously?
a)

```
for (i = 0; i < n; i++)
for (j = 0; j < n; j += 5)
```

b)

```
for (i = 0, j = 0; i < n, j < n; i++, j += 5)
```

c)

```
for (i = 0; i < n;i++){}
for (j = 0; j < n;j += 5){}
```

d) none of the mentioned

ANSWER:- b
Explanation: None.

3. Which for loop has range of similar indexes of ‘i’ used in for (i = 0;i < n; i++)?
a) for (i = n; i>0; i–)
b) for (i = n; i >= 0; i–)
c) for (i = n-1; i>0; i–)
d) for (i = n-1; i>-1; i–)

ANSWER:- d
Explanation: None.

4. Which of the following cannot be used as LHS of the expression in for (exp1;exp2; exp3)?
a) variable
b) function
c) typedef
d) macros

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        short i;
5.        for (i = 1; i >= 0; i++)
6.            printf ("%d\n", i);
7.
8.    }
```

a) The control won’t fall into the for loop
b) Numbers will be displayed until the signed limit of short and throw a runtime error
c) Numbers will be displayed until the signed limit of short and program will successfully terminate
d) This program will get into an infinite loop and keep printing numbers with no errors

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int k = 0;
5.        for (k)
6.            printf ("Hello");
7.    }
```

a) Compile time error
b) hello
c) Nothing
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int k = 0;
5.        for (k < 3; k++)
6.        printf ("Hello");
7.    }
```

a) Compile time error
b) Hello is printed thrice
c) Nothing
d) Varies

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        double k = 0;
5.        for (k = 0.0; k < 3.0;
    k++)
6.            printf ("Hello");
7.    }
```

a) Run time error
b) Hello is printed thrice
c) Hello is printed twice
d) Hello is printed infinitely

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
```

```
4.          double k = 0;
5.          for (k = 0.0; k < 3.0;
    k++);
6.              printf ("%lf", k);
7.      }
```

a) 2.000000
b) 4.000000
c) 3.000000
d) Run time error

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          int k;
5.          for (k = -3; k < -5; k++)
6.              printf ("Hello");
7.      }
```

a) Hello
b) Infinite hello
c) Run time error
d) Nothing

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int i = 0;
5.          for (; ; ;)
6.              printf ("In for
    loop\n");
7.              printf ("After
    loop\n");
8.      }
```

a) Compile time error
b) Infinite loop
c) After loop
d) Undefined behaviour

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int i = 0;
5.          for (i++; i == 1; i = 2)
6.              printf ("In for loop
    ");
7.              printf ("After
    loop\n");
8.      }
```

a) In for loop after loop
b) After loop
c) Compile time error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int i = 0;
5.          for (foo(); i == 1; i =
    2)
6.              printf ("In for
    loop\n");
7.              printf ("After
    loop\n");
8.      }
9.      int foo()
10.      {
11.          return 1;
12.      }
```

a) After loop
b) In for loop after loop
c) Compile time error
d) Infinite loop

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.      #include <stdio.h>
```

```
2.    int main()
3.    {
4.        int *p = NULL;
5.        for (foo(); p; p = 0)
6.            printf ("In for
    loop\n");
7.            printf ("After
    loop\n");
8.    }
```

a) In for loop after loop
b) Compile time error
c) Infinite loop
d) Depends on the value of NULL

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        for (int i = 0;i < 1;
    i++)
5.            printf ("In for
    loop\n");
6.    }
```

a) Compile time error
b) In for loop
c) Depends on the standard compiler implements
d) Depends on the compiler

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        while ()
5.            printf ("In while
    loop ");
6.        printf ("After loop\n");
7.    }
```

a) In while loop after loop
b) After loop
c) Compile time error
d) Infinite loop

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        do
5.            printf ("In while
    loop ");
6.        while (0);
7.            printf ("After
    loop\n");
8.    }
```

a) In while loop
b) In while loop After loop
c) After loop
d) Infinite loop

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 0;
5.        do {
6.            i++;
7.            printf ("In while
    loop\n");
8.        } while (i < 3);
9.    }
```

a)

In while loop

In while loop

In while loop

b)

In while loop

In while loop

c) Depends on the compiler
d) Compile time error

ANSWER:- a
Explanation: None.

4. How many times i value is checked in the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    do {
        i++;
        printf ("in while loop\n");
    } while (i < 3);
}
```

a) 2
b) 3
c) 4
d) 1

ANSWER:- b
Explanation: None.

5. How many times i value is checked in the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    while (i < 3)
        i++;
    printf ("In while loop\n");
}
```

a) 2
b) 3
c) 4
d) 1

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 2;
    do
    {
        printf ("Hi");
    } while (i < 2)
}
```

a) Compile time error
b) Hi Hi
c) Hi
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0;
    while (++i)
    {
        printf ("H");
    }
}
```

a) H
b) H is printed infinite times
c) Compile time error
d) Varies

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0;
    do
    {
        printf ("Hello");
    } while (i != 0);
}
```

a) Nothing
b) H is printed infinite times
c) Hello
d) Run time error

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *str = "";
    do
    {
        printf ("hello");
    } while (str);
}
```

a) Nothing
b) Run time error
c) Varies
d) Hello is printed infinite times

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0;
    while (i < 10)
    {
        i++;
        printf ("hi\n");
        while (i < 8)
         {
            i++;
            printf ("hello\n");
         }
     }
}
```

a) Hi is printed 8 times, hello 7 times and then hi 2 times
b) Hi is printed 10 times, hello 7 times
c) Hi is printed once, hello 7 times
d) Hi is printed once, hello 7 times and then hi 2 times

ANSWER:- d
Explanation: None.

3. What is an example of iteration in C?
a) for
b) while
c) do-while
d) all of the mentioned

ANSWER:- d
Explanation: None.

4. How many times while loop condition is tested in the following C code snippets, if i is initialized to 0 in both the cases?

```
while (i < n)
        i++;
    ———-
    do
        i++;
    while (i <= n);
```

a) n, n
b) n, n+1
c) n+1, n
d) n+1, n+1

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    while (i = 0)
        printf ("True\n");
    printf ("False\n");
}
```

a) True (infinite time)
b) True (1 time) False
c) False
d) Compiler dependent

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    while (i < 5, j < 10)
    {
        i++;
        j++;
    }
     printf ("%d, %d\n", i, j);
}
```

a) 5, 5
b) 5, 10
c) 10, 10
d) Syntax error

ANSWER:- c
Explanation: None.

7. Which loop is most suitable to first perform the operation and then test the condition?
a) for loop
b) while loop
c) do-while loop
d) none of the mentioned

ANSWER:- c
Explanation: None.

1. Which keyword can be used for coming out of recursion?
a) break
b) return
c) exit
d) both break and return

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 0, i = 0, b;
    for (i = 0;i < 5; i++)
    {
        a++;
        continue;
    }
}
```

a) 2
b) 3
c) 4
d) 5

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a = 0, i = 0, b;
    for (i = 0;i < 5; i++)
    {
        a++;
        if (i == 3)
            break;
     }
}
```

a) 1
b) 2
c) 3
d) 4

ANSWER:- d
Explanation: None.

4. The keyword 'break' cannot be simply used within _____
a) do-while
b) if-else
c) for
d) while

ANSWER:- b
Explanation: None.

5. Which keyword is used to come out of a loop only for that iteration?
a) break
b) continue
c) return
d) none of the mentioned

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
```

```
2.    void main()
3.    {
4.        int i = 0, j = 0;
5.        for (i = 0;i < 5; i++)
6.        {
7.            for (j = 0;j < 4;
    j++)
8.            {
9.                if (i > 1)
10.                    break;
11.            }
12.            printf ("Hi \n");
13.        }
14.    }
```

a) Hi is printed 5 times
b) Hi is printed 9 times
c) Hi is printed 7 times
d) Hi is printed 4 times

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int i = 0;
5.        int j = 0;
6.        for (i = 0;i < 5; i++)
7.        {
8.            for (j = 0;j < 4;
    j++)
9.            {
10.                if (i > 1)
11.                    continue;
12.                    printf ("Hi
    \n");
13.            }
14.        }
15.    }
```

a) Hi is printed 9 times
b) Hi is printed 8 times
c) Hi is printed 7 times
d) Hi is printed 6 times

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int i = 0;
5.        for (i = 0;i < 5; i++)
6.            if (i < 4)
7.            {
8.                printf ("Hello");
9.                break;
10.            }
11.    }
```

a) Hello is printed 5 times
b) Hello is printed 4 times
c) Hello
d) Hello is printed 3 times

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int i = 0;
5.        if (i == 0)
6.        {
7.            printf ("Hello");
8.            continue;
9.        }
10.    }
```

a) Hello is printed infinite times
b) Hello
c) Varies
d) Compile time error

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int i = 0;
5.        if (i == 0)
```

```
        {
            printf ("Hello");
            break;
        }
    }
```

a) Hello is printed infinite times
b) Hello
c) Varies
d) Compile time error

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    do
    {
        i++;
        if (i == 2)
            continue;
             printf ("In while loop ");
     } while (i < 2);
     printf ("%d\n", i);
}
```

a) In while loop 2
b) In while loop in while loop 3
c) In while loop 3
d) Infinite loop

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    for (i; i < 2; i++){
        for (j = 0; j < 3; j++)
        {
            printf ("1\n");
            break;
        }
        printf ("2\n");
    }
    printf ("after loop\n");
}
```

a)

```
1
2
after loop
```

b)

```
1
after loop
```

c)

```
1
2
1
2
after loop
```

d)

```
1
1
2
after loop
```

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0;
    while (i < 2)
    {
        if (i == 1)
            break;
            i++;
```

```
10.                 if (i == 1)
11.                     continue;
12.                     printf ("In
    while loop\n");
13.         }
14.         printf ("After loop\n");
15.     }
```

a)

In while loop

After loop

b) After loop
c)

In while loop

In while loop

After loop

d) In while loop

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         int i = 0;
5.         char c = 'a';
6.         while (i < 2)
7.         {
8.             i++;
9.             switch (c)
10.             {
11.                 case 'a':
12.                     printf ("%c
    ", c);
13.                     break;
14.                     break;
15.             }
16.         }
17.         printf ("after loop\n");
18.     }
```

a) a after loop
b) a a after loop
c) after loop
d) error

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         printf ("before continue
    ");
5.         continue;
6.         printf ("after
    continue\n");
7.     }
```

a) Before continue after continue
b) Before continue
c) After continue
d) Compile time error

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         printf ("%d ", 1);
5.         goto l1;
6.         printf ("%d ", 2);
7.         l1:goto l2;
8.         printf ("%d ", 3);
9.         l2:printf ("%d ", 4);
10.    }
```

a) 1 4
b) Compilation error
c) 1 2 4
d) 1 3 4

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    printf ("%d ", 1);
    l1:l2:
    printf ("%d ", 2);
    printf ("%d\n", 3);
}
```

a) Compilation error
b) 1 2 3
c) 1 2
d) 1 3

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    printf ("%d ", 1);
    goto l1;
    printf ("%d ", 2);
}
void foo()
{
     l1 : printf ("3 ", 3);
 }
```

a) 1 2 3
b) 1 3
c) 1 3 2
d) Compilation error

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    while (i < 2)
    {
        l1 : i++;
        while (j < 3)
        {
            printf ("Loop\n");
            goto l1;
        }
    }
}
```

a) Loop Loop
b) Compilation error
c) Loop Loop Loop Loop
d) Infinite Loop

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    while (l1: i < 2)
    {
        i++;
        while (j < 3)
        {
            printf ("loop\n");
            goto l1;
        }
    }
}
```

a) loop loop
b) Compilation error
c) loop loop loop loop
d) Infinite loop

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    l1: while (i < 2)
    {
        i++;
```

```
            while (j < 3)
            {
                printf
   ("loop\n");
                goto l1;
             }
        }
    }
```

a) loop loop
b) compilation error
c) oop loop loop loop
d) infinite loop

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0;
    if (i == 0)
    {
        goto label;
    }
    label: printf ("Hello");
}
```

a) Nothing
b) Error
c) Infinite Hello
d) Hello

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0, k;
    if (i == 0)
        goto label;
        for (k = 0;k < 3;
   k++)
        {
            printf ("hi\n");
            label: k =
   printf ("%03d", i);
        }
    }
```

a) 0
b) hi hi hi 0 0 0
c) 0 hi hi hi 0 0 0
d) 0 0 0

ANSWER:- a
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 0, k;
    label: printf ("%d", i);
    if (i == 0)
        goto label;
}
```

a) 0
b) Infinite 0
c) Nothing
d) Error

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int i = 5, k;
    if (i == 0)
        goto label;
        label: printf ("%d",
   i);
        printf ("Hey");
}
```

a) 5
b) Hey
c) 5 Hey
d) Nothing

ANSWER:- c
Explanation: None.

2. goto can be used to jump from main() to within a function.
a) true
b) false
c) depends
d) varies

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    printf ("%d ", 1);
    goto l1;
    printf ("%d ", 2);
    l1:goto l2;
    printf ("%d ", 3);
    l2:printf ("%d ", 4);
}
```

a) 1 4
b) Compile time error
c) 1 2 4
d) 1 3 4

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    printf ("%d ", 1);
    l1:l2:
    printf ("%d ", 2);
    printf ("%d\n", 3);
}
```

a) Compile time error
b) 1 2 3
c) 1 2
d) 1 3

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    printf ("%d ", 1);
    goto l1;
    printf ("%d ", 2);
}
void foo()
{
     l1: printf ("3 ", 3);
 }
```

a) 1 2 3
b) 1 3
c) 1 3 2
d) Compile time error

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    while (i < 2)
    {
        l1: i++;
        while (j < 3)
        {
            printf
   ("loop\n");
            goto l1;
         }
     }
}
```

a) loop loop
b) Compile time error
c) loop loop loop loop
d) Infinite loop

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
```

```
    {
        int i = 0, j = 0;
        while (l1: i < 2)
        {
            i++;
            while (j < 3)
            {
                printf
("loop\n");
                goto l1;
            }
        }
    }
```

a) loop loop
b) Compile time error
c) loop loop loop loop
d) Infinite loop

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 0;
    l1: while (i < 2)
        {
            i++;
            while (j < 3)
            {
                printf
("loop\n");
                goto l1;
            }
        }
}
```

a) loop loop
b) Compile time error
c) loop loop loop loop
d) Infinite loop

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    void foo();
    printf("1 ");
    foo();
}
void foo()
{
     printf("2 ");
 }
```

a) 1 2
b) Compile time error
c) 1 2 1 2
d) Depends on the compiler

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    void foo(), f();
    f();
}
void foo()
{
    printf("2 ");
 }
 void f()
 {
     printf("1 ");
     foo();
 }
```

a) Compile time error as foo is local to main
b) 1 2
c) 2 1
d) Compile time error due to declaration of functions inside main

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
```

```
3.    {
4.        void foo();
5.        void f()
6.        {
7.            foo();
8.        }
9.        f();
10.    }
11.    void foo()
12.    {
13.        printf("2 ");
14.    }
```

a) 2 2
b) 2
c) Compile time error
d) Depends on the compiler

ANSWER:- d
Explanation: Even though the answer is 2, this code will compile fine only with gcc. GNU C supports nesting of functions in C as a language extension whereas standard C compiler doesn't.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void foo();
3.    int main()
4.    {
5.        void foo();
6.        foo();
7.        return 0;
8.    }
9.    void foo()
10.    {
11.        printf("2 ");
12.    }
```

a) Compile time error
b) 2
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void foo();
3.    int main()
4.    {
5.        void foo(int);
6.        foo(1);
7.        return 0;
8.    }
9.    void foo(int i)
10.    {
11.        printf("2 ");
12.    }
```

a) 2
b) Compile time error
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void foo();
3.    int main()
4.    {
5.        void foo(int);
6.        foo();
7.        return 0;
8.    }
9.    void foo()
10.    {
11.        printf("2 ");
12.    }
```

a) 2
b) Compile time error
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m()
3.    {
4.        printf("hi");
5.    }
6.    void main()
```

```
7.      {
8.          m();
9.      }
```

a) hi
b) Run time error
c) Nothing
d) Varies

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void m();
3.      void n()
4.      {
5.          m();
6.      }
7.      void main()
8.      {
9.          void m()
10.         {
11.             printf("hi");
12.         }
13.     }
```

a) hi
b) Compile time error
c) Nothing
d) Varies

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          m();
5.          void m()
6.          {
7.              printf("hi");
8.          }
9.      }
```

a) hi
b) Compile time error
c) Nothing
d) Varies

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          m();
5.      }
6.      void m()
7.      {
8.          printf("hi");
9.          m();
10.     }
```

a) Compile time error
b) hi
c) Infinite hi
d) Nothing

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          static int x = 3;
5.          x++;
6.          if (x <= 5)
7.          {
8.              printf("hi");
9.              main();
10.         }
11.     }
```

a) Run time error
b) hi
c) Infinite hi
d) hi hi

ANSWER:- d
Explanation: None.

4. Which of the following is a correct format for declaration of function?
a) return-type function-name(argument type);
b) return-type function-name(argument type){}
c) return-type (argument type)function-name;

d) all of the mentioned

ANSWER:- a
Explanation: None.

5. Which of the following function declaration is illegal?
a) int 1bhk(int);
b) int 1bhk(int a);
c) int 2bhk(int*, int []);
d) all of the mentioned

ANSWER:- d
Explanation: None.

6. Which function definition will run correctly?
a)

```
int sum(int a, int b)
return (a + b);
```

b)

```
int sum(int a, int b)
{return (a + b);}
```

c)

```
int sum(a, b)
return (a + b);
```

d) none of the mentioned

ANSWER:- b
Explanation: None.

7. Can we use a function as a parameter of another function? [E.g.: void wow(int func())].
a) Yes, and we can use the function value conveniently
b) Yes, but we call the function again to get the value, not as convenient as in using variable
c) No, C does not support it
d) This case is compiler dependent

ANSWER:- c
Explanation: None.

8. The value obtained in the function is given back to main by using _____ keyword.
a) return
b) static
c) new
d) volatile

ANSWER:- a
Explanation: None.

1. What is the return-type of the function sqrt()?
a) int
b) float
c) double
d) depends on the data type of the parameter

ANSWER:- c
Explanation: None.

2. Which of the following function declaration is illegal?
a)

```
double func();
int main(){}
double func(){}
```

b)

```
double func(){};
int main(){}
```

c)

```
int main()
{
    double func();
}
double func(){//statements}
```

d) None of the mentioned

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code having void return-type function?

```
1.  #include <stdio.h>
2.  void foo()
3.  {
4.      return 1;
5.  }
6.  void main()
7.  {
8.      int x = 0;
9.      x = foo();
10.     printf("%d", x);
11.  }
```

a) 1
b) 0
c) Runtime error
d) Compile time error

ANSWER:- d
Explanation: None.

4. What will be the data type returned for the following C function?

```
#include <stdio.h>
int func()
{
    return (double)(char)5.0;
}
```

a) char
b) int
c) double
d) multiple type-casting in return is illegal

ANSWER:- b
Explanation: None.

5. What is the problem in the following C declarations?

```
int func(int);
double func(int);
int func(float);
```

a) A function with same name cannot have different signatures
b) A function with same name cannot have different return types
c) A function with same name cannot have different number of parameters
d) All of the mentioned

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int m()
{
    printf("hello");
}
void main()
{
    int k = m();
    printf("%d", k);
}
```

a) hello5
b) Error
c) Nothing
d) Junk value

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int *m()
{
    int *p = 5;
    return p;
}
void main()
{
    int *k = m();
     printf("%d", k);
}
```

a) 5
b) Junk value
c) 0
d) Error

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int *m();
void main()
{
    int *k = m();
    printf("hello ");
    printf("%d", k[0]);
}
int *m()
 {
     int a[2] = {5, 8};
     return a;
 }
```

a) hello 5 8
b) hello 5
c) hello followed by garbage value

d) Compilation error

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int *m();
void main()
{
    int k = m();
    printf("%d", k);
}
int *m()
{
     int a[2] = {5, 8};
     return a;
 }
```

a) 5
b) 8
c) Nothing
d) Varies

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void m(int k)
{
    printf("hi");
}
void m(double k)
{
    printf("hello");
}
 void main()
 {
     m(3);
 }
```

a) hi
b) hello
c) Compile time error
d) Nothing

ANSWER:- c
Explanation: None.

3. What is the default return type if it is not specified in function definition?
a) void
b) int
c) double
d) short int

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int foo();
int main()
{
    int i = foo();
}
foo()
{
    printf("2 ");
     return 2;
 }
```

a) 2
b) Compile time error
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
double foo();
int main()
{
    foo();
    return 0;
}
foo()
{
     printf("2 ");
     return 2;
 }
```

a) 2
b) Compile time error
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

6. Functions can return structure in C?
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

7. Functions can return enumeration constants in C?
a) true
b) false
c) depends on the compiler
d) depends on the standard

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
enum m{JAN, FEB, MAR};
enum m foo();
int main()
{
    enum m i = foo();
    printf("%d\n", i);
}
int  foo()
 {
     return JAN;
 }
```

a) Compile time error
b) 0
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    m();
    printf("%d", x);
}
int x;
void m()
{
     x = 4;
 }
```

a) 4
b) Compile time error
c) 0
d) Undefined

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int x;
void main()
{
    printf("%d", x);
}
```

a) Junk value
b) Run time error
c) 0
d) Undefined

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int x = 5;
void main()
{
    int x = 3;
    printf("%d", x);
    {
        x = 4;
    }
     printf("%d", x);
 }
```

a) Run time error
b) 3 3
c) 3 5
d) 3 4

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int x = 5;
void main()
{
    int x = 3;
    printf("%d", x);
    {
        int x = 4;
    }
     printf("%d", x);
 }
```

a) 3 3
b) 3 4
c) 3 5
d) Run time error

ANSWER:- a
Explanation: None.

5. Functions in C are always _____
a) Internal
b) External
c) Both Internal and External
d) External and Internal are not valid terms for functions

ANSWER:- b
Explanation: None.

6. Global variables are _____
a) Internal
b) External
c) Both Internal and External
d) None of the mentioned

ANSWER:- b
Explanation: None.

7. Which of the following is an external variable in the following C code?

```
#include <stdio.h>
int func (int a)
{
    int b;
    return b;
}
int main()
{
    int c;
     func (c);
 }
int d;
```

a) a
b) b
c) c
d) d

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
 #include <stdio.h>
int main()
{
    printf("%d", d++);
}
int d = 10;
```

a) 9
b) 10
c) 11
d) Compile time error

ANSWER:- d
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
double var = 8;
int main()
{
    int var = 5;
    printf("%d", var);
}
```

a) 5
b) 8
c) Compile time error due to wrong format identifier for double
d) Compile time error due to redeclaration of variable with same name

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```c
#include <stdio.h>
double i;
int main()
{
   printf("%g\n",i);
   return 0;
}
```

a) 0
b) 0.000000
c) Garbage value
d) Depends on the compiler

ANSWER:- a
Explanation: None.

2. Which part of the program address space is p stored in the following C code?

```c
#include <stdio.h>
int *p = NULL;
int main()
{
    int i = 0;
    p = &i;
    return 0;
}
```

a) Code/text segment
b) Data segment
c) Bss segment
d) Stack

ANSWER:- b
Explanation: None.

3. Which part of the program address space is p stored in the following C code?

```c
#include <stdio.h>
int *p;
int main()
{
    int i = 0;
    p = &i;
    return 0;
}
```

a) Code/text segment
b) Data segment
c) Bss segment
d) Stack

ANSWER:- c
Explanation: None.

4. Can variable i be accessed by functions in another source file?

```c
#include <stdio.h>
int i;
int main()
{
    printf("%d\n", i);
}
```

a) Yes
b) No
c) Only if static keyword is used
d) Depends on the type of the variable

ANSWER:- a
Explanation: None.

5. Property of the external variable to be accessed by any source file is called by the C90 standard as _____
a) external linkage
b) external scope
c) global scope
d) global linkage

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```c
#include <stdio.h>
int *i;
int main()
{
    if (i == NULL)
        printf("true\n");
    return 0;
}
```

a) true
b) true only if NULL value is 0
c) Compile time error
d) Nothing

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int *i;
int main()
{
    if (i == 0)
        printf("true\n");
    return 0;
}
```

a) true
b) true only if NULL value is 0
c) Compile time error
d) Nothing

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
static int x = 5;
void main()
{
    x = 9;
    {
        int x = 4;
    }
    printf("%d", x);
 }
```

a) 9
b) 4
c) 5
d) 0

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int i;
int main()
{
    extern int i;
    if (i == 0)
        printf("scope
rules\n");
}
```

a) scope rules
b) Compile time error due to multiple declaration
c) Compile time error due to not defining type in statement extern i
d) Nothing will be printed as value of i is not zero because i is an automatic variable

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code (without linking the source file in which ary1 is defined)?

```
#include <stdio.h>
int main()
{
    extern ary1[];
    printf("scope rules\n");
}
```

a) scope rules
b) Linking error due to undefined reference
c) Compile time error because size of array is not provided
d) Compile time error because datatype of array is not provided

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code (after linking to source file having definition of ary1)?

```
#include <stdio.h>
int main()
{
    extern ary1[];
    printf("%d\n", ary1[0]);
}
```

a) Value of ary1[0];
b) Compile time error due to multiple definition
c) Compile time error because size of array is not provided
d) Compile time error because datatype of array is not provided

ANSWER:- d
Explanation: None.

4. What is the scope of an external variable?
a) Whole source file in which it is defined
b) From the point of declaration to the end of the

file in which it is defined
c) Any source file in a program
d) From the point of declaration to the end of the file being compiled

ANSWER:- d
Explanation: None.

5. What is the scope of a function?
a) Whole source file in which it is defined
b) From the point of declaration to the end of the file in which it is defined
c) Any source file in a program
d) From the point of declaration to the end of the file being compiled

ANSWER:- d
Explanation: None.

6. Comment on the output of the following C code.

```
#include <stdio.h>
int main()
{
    int i;
    for (i = 0;i < 5; i++)
    int a = i;
    printf("%d", a);
}
```

a) a is out of scope when printf is called
b) Redeclaration of a in same scope throws error
c) Syntax error in declaration of a
d) No errors, program will show the output 5

ANSWER:- c
Explanation: None.

7. Which variable has the longest scope in the following C code?

```
#include <stdio.h>
int b;
int main()
{
    int c;
    return 0;
}
int a;
```

a) a
b) b
c) c
d) Both a and b

ANSWER:- b
Explanation: None.

8. Comment on the following 2 C programs.

```
#include <stdio.h> //Program 1
int main()
{
    int a;
    int b;
    int c;
}

#include <stdio.h> //Program 2
 int main()
 {
     int a;
     {
         int b;
     }
     {
         int c;
     }
 }
```

a) Both are same
b) Scope of c is till the end of the main function in Program 2
c) In Program 1, variables a, b and c can be used anywhere in the main function whereas in Program 2, variables b and c can be used only inside their respective blocks
d) None of the mentioned

ANSWER:- c
Explanation: None.

1. What will be the sequence of allocation and deletion of variables in the following C code?

```
#include <stdio.h>
int main()
{
    int a;
    {
        int b;
```

```
        }
    }
```

a) a->b, a->b
b) a->b, b->a
c) b->a, a->b
d) b->a, b->a

ANSWER:- b
Explanation: None.

2. Array sizes are optional during array declaration by using _____ keyword.
a) auto
b) static
c) extern
d) register

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 3;
    {
        x = 4;
        printf("%d", x);
    }
}
```

a) 4
b) 3
c) 0
d) Undefined

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int x = 5;
void main()
{
    int x = 3;
    m();
    printf("%d", x);
}
void m()
 {
    x = 8;
    n();
 }
void n()
{
    printf("%d", x);
}
```

a) 8 3
b) 3 8
c) 8 5
d) 5 3

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int x;
void main()
{
    m();
    printf("%d", x);
}
void m()
{
    x = 4;
 }
```

a) 0
b) 4
c) Compile time error
d) Undefined

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
static int x = 5;
void main()
{
    int x = 9;
    {
        x = 4;
    }
    printf("%d", x);
 }
```

a) 9
b) 5
c) 4
d) 0

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    {
        int x = 8;
    }
    printf("%d", x);
}
```

a) 8
b) 0
c) Undefined
d) Compile time error

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    m();
    m();
}
void m()
{
    static int x = 5;
     x++;
     printf("%d", x);
 }
```

a) 6 7
b) 6 6
c) 5 5
d) 5 6

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    static int x;
    printf("x is %d", x);
}
```

a) 0
b) 1
c) Junk value
d) Run time error

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
static int x;
void main()
{
    int x;
    printf("x is %d", x);
}
```

a) 0
b) Junkvalue
c) Run time error
d) Nothing

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    static double x;
    int x;
    printf("x is %d", x);
}
```

a) Nothing
b) 0
c) Compile time error
d) Junkvalue

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    static int x;
    if (x++ < 2)
    main();
}
```

a) Infinite calls to main
b) Run time error
c) Varies
d) main is called twice

ANSWER:- d
Explanation: None.

6. Which of following is not accepted in C?
a) static a = 10; //static as
b) static int func (int); //parameter as static
c) static static int a; //a static variable prefixed with static
d) all of the mentioned

ANSWER:- c
Explanation: None.

7. Which of the following cannot be static in C?
a) Variables
b) Functions
c) Structures
d) None of the mentioned

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code if these two files namely test.c and test1.c are linked and run?

```
-------file test.c-------
#include <stdio.h>
#include ""test.h""
int main()
{
    i = 10;
    printf(""%d "", i);
    foo();
}

-----file test1.c------
#include <stdio.h>
#include ""test.h""
int foo()
{
    printf(""%d\n"", i);
}

-----file test.h-----
#include <stdio.h>
#include <stdlib.h>
static int i;
```

a) 10 0
b) 0 0
c) 10 10
d) Compilation Error

ANSWER:- a
Explanation: None.

2. Functions have static qualifier for its declaration by default.
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

3. Is initialisation mandatory for local static variables?
a) Yes
b) No
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    foo();
    foo();
}
void foo()
{
    int i = 11;
     printf("%d ", i);
     static int j = 12;
     j = j + 1;
     printf("%d\n", j);
```

```
14.     }
```

a) 11 12 11 12
b) 11 13 11 14
c) 11 12 11 13
d) Compile time error

ANSWER:- b
Explanation: None.

5. Assignment statements assigning value to local static variables are executed only once.
a) True
b) False
c) Depends on the code
d) None of the mentioned

ANSWER:- b
Explanation: None.

6. What is the format identifier for “static a = 20.5;”?
a) %s
b) %d
c) %f
d) Illegal declaration due to absence of data type

ANSWER:- b
Explanation: None.

7. Which of the following is true for the static variable?
a) It can be called from another function
b) It exists even after the function ends
c) It can be modified in another function by sending it as a parameter
d) All of the mentioned

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void func();
3.    int main()
4.    {
5.        static int b = 20;
6.        func();
7.    }
8.    void func()
9.    {
10.        static int b;
11.        printf("%d", b);
12.    }
```

a) Output will be 0
b) Output will be 20
c) Output will be a garbage value
d) Compile time error due to redeclaration of static variable

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        register int i = 10;
5.        int *p = &i;
6.        *p = 11;
7.        printf("%d %d\n", i, *p);
8.    }
```

a) Depends on whether i is actually stored in machine register
b) 10 10
c) 11 11
d) Compile time error

ANSWER:- d
Explanation: None.

2. register keyword mandates compiler to place it in machine register.
a) True
b) False
c) Depends on the standard
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        register static int i =
    10;
5.        i = 11;
6.        printf("%d\n", i);
7.    }
```

a) 10
b) Compile time error
c) Undefined behaviour

d) 11

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    register auto int i = 10;
    i = 11;
    printf("%d\n", i);
}
```

a) 10
b) Compile time error
c) Undefined behaviour
d) 11

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    register const int i =
10;
    i = 11;
    printf("%d\n", i);
}
```

a) 10
b) Compile time error
c) Undefined behaviour
d) 11

ANSWER:- b
Explanation: None.

6. Register storage class can be specified to global variables.
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

7. Which among the following is wrong for “register int a;”?
a) Compiler generally ignores the request
b) You cannot take the address of this variable
c) Access time to a is critical
d) None of the mentioned

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    register int x = 5;
    m();
    printf("x is %d", x);
}
void m()
{
     x++;
 }
```

a) 6
b) 5
c) Junk value
d) Compile time error

ANSWER:- d
Explanation: None.

1. When compiler accepts the request to use the variable as a register?
a) It is stored in CPU
b) It is stored in cache memory
c) It is stored in main memory
d) It is stored in secondary memory

ANSWER:- a
Explanation: None.

2. Which data type can be stored in register?
a) int
b) long
c) float
d) all of the mentioned

ANSWER:- d
Explanation: None.

3. Which of the following operation is not possible in a register variable?
a) Reading the value into a register variable
b) Copy the value from a memory variable
c) Global declaration of register variable

d) All of the mentioned

ANSWER:- d
Explanation: None.

4. Which among the following is the correct syntax to declare a static variable register?
a) static register a;
b) register static a;
c) Both static register a; and register static a;
d) We cannot use static and register together

ANSWER:- d
Explanation: None.

5. Register variables reside in _____
a) stack
b) registers
c) heap
d) main memory

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        register int x = 0;
5.        if (x < 2)
6.        {
7.            x++;
8.            main();
9.        }
10.    }
```

a) Segmentation fault
b) main is called twice
c) main is called once
d) main is called thrice

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        register int x;
5.        printf("%d", x);
6.    }
```

a) 0
b) Junk value
c) Compile time error
d) Nothing

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    register int x;
3.    void main()
4.    {
5.        printf("%d", x);
6.    }
```

a) Varies
b) 0
c) Junk value
d) Compile time error

ANSWER:- d
Explanation: None.

1. What is the scope of an automatic variable?
a) Within the block it appears
b) Within the blocks of the block it appears
c) Until the end of program
d) Within the block it appears & Within the blocks of the block it appears

ANSWER:- d
Explanation: None.

2. Automatic variables are allocated space in the form of a _____
a) stack
b) queue
c) priority queue
d) random

ANSWER:- a
Explanation: None.

3. Which of the following is a storage specifier?
a) enum
b) union
c) auto
d) volatile

ANSWER:- c
Explanation: None.

4. If storage class is not specified for a local variable, then the default class will be auto.
a) True

b) False
c) Depends on the standard
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void foo(auto int i);
int main()
{
    foo(10);
}
void foo(auto int i)
{
    printf("%d\n", i );
 }
```

a) 10
b) Compile time error
c) Depends on the standard
d) None of the mentioned

ANSWER:- b
Explanation: None.

6. Automatic variables are stored in _____
a) stack
b) data segment
c) register
d) heap

ANSWER:- a
Explanation: None.

7. What linkage does automatic variables have?
a) Internal linkage
b) External linkage
c) No linkage
d) None of the mentioned

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    auto i = 10;
    const auto int *p = &i;
    printf("%d\n", i);
}
```

a) 10
b) Compile time error
c) Depends on the standard
d) Depends on the compiler

ANSWER:- a
Explanation: None.

1. Automatic variables are _____
a) Declared within the scope of a block, usually a function
b) Declared outside all functions
c) Declared with the auto keyword
d) Declared within the keyword extern

ANSWER:- a
Explanation: None.

2. What is the scope of an automatic variable?
a) Exist only within that scope in which it is declared
b) Cease to exist after the block is exited
c) Exist only within that scope in which it is declared & exist after the block is exited
d) All of the mentioned

ANSWER:- c
Explanation: None.

3. Automatic variables are allocated memory in _____
a) heap
b) Data segment
c) Code segment
d) stack

ANSWER:- d
Explanation: None.

4. What will be the x in the following C code?

```
#include <stdio.h>
void main()
{
    int x;
}
```

a) automatic variable
b) static variable
c) register variable
d) global variable

ANSWER:- a
Explanation: None.

5. Automatic variables are initialized to _____
a) Zero
b) Junk value
c) Nothing
d) Both Zero & Junk value

ANSWER:- b
Explanation: None.

6. Which of the following storage class supports char data type?
a) register
b) static
c) auto
d) all of the mentioned

ANSWER:- d
Explanation: None.

7. A local variable declaration with no storage class specified is by default _____
a) auto
b) extern
c) static
d) register

ANSWER:- a
Explanation: None.

1. Property which allows to produce different executable for different platforms in C is called?
a) File inclusion
b) Selective inclusion
c) Conditional compilation
d) Recursive macros

ANSWER:- c
Explanation: Conditional compilation is the preprocessor facility to produce a different executable.

2. What is #include <stdio.h>?
a) Preprocessor directive
b) Inclusion directive
c) File inclusion directive
d) None of the mentioned

ANSWER:- a
Explanation: None.

3. C preprocessors can have compiler specific features.
a) True
b) False
c) Depends on the standard
d) Depends on the platform

ANSWER:- a
Explanation: #pragma is compiler specific feature.

4. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(m, n) m * n = 10
3. int main()
4. {
5.     printf("in main\n");
6. }
```

a) In main
b) Compilation error as lvalue is required for the expression m*n=10
c) Preprocessor error as lvalue is required for the expression m*n=10
d) None of the mentioned

ANSWER:- a
Explanation: Preprocessor just replaces whatever is given compiler then checks for error at the replaced part of the code. Here it is not replaced anywhere.
Output:
$ cc pgm1.c
$ a.out
in main

5. C preprocessor is conceptually the first step during compilation.
a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

6. Preprocessor feature that supply line numbers and filenames to compiler is called?
a) Selective inclusion
b) macro substitution
c) Concatenation
d) Line control

ANSWER:- d
Explanation: None.

7. #include <somefile.h> are _____ files and #include “somefile.h” _____ files.
a) Library, Library
b) Library, user-created header
c) User-created header, library
d) They can include all types of file

ANSWER:- d
Explanation: Both of these statement can be used to select any file.

8. What is a preprocessor?
a) That processes its input data to produce output that is used as input to another program
b) That is nothing but a loader
c) That links various source files
d) All of the mentioned

ANSWER:- a
Explanation: A preprocessor is a program that processes its input data to produce output that is used as input to another program.

1. Which of the following are C preprocessors?
a) #ifdef
b) #define
c) #endif
d) all of the mentioned

ANSWER:- d
Explanation: None.

2. #include statement must be written _____
a) Before main()
b) Before any scanf/printf
c) After main()
d) It can be written anywhere

ANSWER:- a
Explanation: Using these directives before main() improves readability.

3. #pragma exit is primarily used for?
a) Checking memory leaks after exiting the program
b) Informing Operating System that program has terminated
c) Running a function at exiting the program
d) No such preprocessor exist

ANSWER:- c
Explanation: It is primarily used for running a function upon exiting the program.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int one = 1, two = 2;
5.        #ifdef next
6.        one = 2;
7.        two = 1;
8.        #endif
9.        printf("%d, %d", one,
    two);
10.    }
```

a) 1, 1
b) 1, 2
c) 2, 1
d) 2, 2

ANSWER:- b
Explanation: None.

5. The C-preprocessors are specified with _____symbol.
a) #
b) $
c) ” ”
d) &

ANSWER:- a
Explanation: The C-preprocessors are specified with # symbol.

6. What is #include directive?
a) Tells the preprocessor to grab the text of a file and place it directly into the current file
b) Statements are not typically placed at the top of a program
c) All of the mentioned
d) None of the mentioned

ANSWER:- a
Explanation: The #include directive tells the preprocessor to grab the text of a file and place it directly into the current file and are statements are typically placed at the top of a program.

7. The preprocessor provides the ability for _____
a) The inclusion of header files
b) The inclusion of macro expansions
c) Conditional compilation and line control
d) All of the mentioned

ANSWER:- d
Explanation: The preprocessor provides the ability for the inclusion of header files, macro expansions, conditional compilation, and line control.

8. If #include is used with file name in angular brackets.
a) The file is searched for in the standard compiler include paths
b) The search path is expanded to include the current source directory

c) The search path will expand
d) None of the mentioned

ANSWER:- a
Explanation: With the #include, if the filename is enclosed within angle brackets, the file is searched for in the standard compiler include paths.

1. What is the sequence for preprocessor to look for the file within <>?
a) The predefined location then the current directory
b) The current directory then the predefined location
c) The predefined location only
d) The current directory location

ANSWER:- a
Explanation: <> first searches the predefined location for the specified file and then the current directory.

2. Which directory the compiler first looks for the file when using #include?
a) Current directory where program is saved
b) C:COMPILERINCLUDE
c) S:SOURCEHEADERS
d) Both C:COMPILERINCLUDE and S:SOURCEHEADERS simultaneously

ANSWER:- b
Explanation: None.

3. What would happen if you create a file stdio.h and use #include “stdio.h”?
a) The predefined library file will be selected
b) The user-defined library file will be selected
c) Both the files will be included
d) The compiler won’t accept the program

ANSWER:- b
Explanation: None.

4. How is search done in #include and #include “somelibrary.h” according to C standard?
a) When former is used, current directory is searched and when latter is used, standard directory is searched
b) When former is used, standard directory is searched and when latter is used, current directory is searched
c) When former is used, search is done in implementation defined manner and when latter is used, current directory is searched
d) For both, search for ‘somelibrary’ is done in implementation-defined places

ANSWER:- b
Explanation: None.

5. How is search done in #include and #include”somelibrary.h” normally or conventionally?
a) When former is used, current directory is searched and when latter is used, standard directory is searched
b) When former is used, predefined directory is searched and when latter is used, current directory is searched and then predefined directories are searched
c) When former is used, search is done in implementation defined manner and latter is used to search current directory
d) For both, search for somelibrary is done in implementation-defined manner

ANSWER:- b
Explanation: None.

6. Can function definition be present in header files?
a) Yes
b) No
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

7. Comment on the output of the following C code.

```
1.    #include <stdio.h>
2.    #include "test.h"
3.    #include "test.h"
4.    int main()
5.    {
6.        //some code
7.    }
```

a) True
b) Compile time error
c) False
d) Depends on the compiler

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #define foo(m, n) m ## n
```

```
3.     void myfunc();
4.     int main()
5.     {
6.         myfunc();
7.     }
8.     void myfunc()
9.     {
10.         printf("%d\n", foo(2,
    3));
11.     }
```

a) 23
b) 2 3
c) Compile time error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

1. If the file name is enclosed in double quotation marks, then _____
a) The preprocessor treats it as a user-defined file
b) The preprocessor treats it as a system-defined file
c) The preprocessor treats it as a user-defined file & system-defined file
d) None of the mentioned

ANSWER:- a
Explanation: None.

2. If the file name is enclosed in angle brackets, then _____
a) The preprocessor treats it as a user-defined file
b) The preprocessor treats it as a system-defined file
c) The preprocessor treats it as a user-defined file & system-defined file
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code snippet?

```
1.     #include (stdio.h)
2.     void main()
3.     {
4.         printf("hello");
5.     }
```

a) hello
b) Nothing
c) Compile time error
d) Depends on compiler

ANSWER:- c
Explanation: File to be included must be specified either in “” or <>.
Output:
$ cc pgm1.c
pgm1.c:1: error: #include expects “FILENAME” or
pgm1.c: In function ‘main’:
pgm1.c:4: warning: incompatible implicit declaration of built-in function ‘printf’

4. The below two lines are equivalent to _____

```
1.     #define C_IO_HEADER
    <stdio.h>
2.     #include C_IO_HEADER
```

a) #include<stdlib.h>
b) #include”printf”
c) #include”C_IO_HEADER”
d) #include<stdio.h>

ANSWER:- d
Explanation: Since C_IO_HEADER is defined to be <stdio.h>, the second line becomes #include<stdio.h>, since C_IO_HEADER is replaced with <stdio.h>

5. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     #include "printf"
3.     void main()
4.     {
5.         printf("hello");
6.     }
```

a) hello
b) Error
c) Depends on compiler
d) Varies

ANSWER:- b
Explanation: None.

6. Which of the following file extensions are accepted with #include?
a) .h
b) .in
c) .com

d) All of the mentioned

ANSWER:- d
Explanation: The preprocessor will include whatever file extension you specify in your #include statement. However, it is not a good practice as another person debugging it will find it difficult in finding files you have included.

7. Which of the following names for files not accepted?
a) header.h.h
b) 123header.h
c) _head_er.h
d) None of the mentioned

ANSWER:- d
Explanation: All file names are accepted as for the execution to occur. There are no constraints on giving file names for inclusion.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #define foo(m, n) m ## n
3.    int main()
4.    {
5.        printf("%s\n", foo(k,
   l));
6.    }
```

a) k l
b) kl
c) Compile time error
d) Undefined behaviour

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #define foo(m, n) " m ## n "
3.    int main()
4.    {
5.        printf("%s\n", foo(k,
   l));
6.    }
```

a) k l
b) kl
c) Compile time error
d) m ## n

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #define foo(x, y) #x #y
3.    int main()
4.    {
5.        printf("%s\n", foo(k,
   l));
6.        return 0;
7.    }
```

a) kl
b) k l
c) xy
d) Compile time error

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #define foo(x, y) x / y + x
3.    int main()
4.    {
5.        int i = -6, j = 3;
6.        printf("%d\n",foo(i + j,
   3));
7.        return 0;
8.    }
```

a) Divided by zero exception
b) Compile time error
c) -8
d) -4

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f();
3.    int main()
4.    {
5.        #define foo(x, y) x / y
   + x
6.        f();
```

```
    }
    void f()
    {
        printf("%d\n", foo(-3,
3));
     }
```

a) -8
b) -4
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void f();
int main()
{
    #define max 10
    f();
    return 0;
}
void f()
 {
     printf("%d\n", max *
10);
 }
```

a) 100
b) Compile time error since #define cannot be inside functions
c) Compile time error since max is not visible in f()
d) Undefined behaviour

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
#define foo(x, y) x / y + x
int main()
{
    int i = -6, j = 3;
    printf("%d ", foo(i + j,
3));
    printf("%d\n", foo(-3,
3));
    return 0;
}
```

a) -8 -4
b) -4 divided by zero exception
c) -4 -4
d) Divided by zero exception

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
    int foo(int, int);
    #define foo(x, y) x / y + x
    int main()
    {
        int i = -6, j = 3;
        printf("%d ",foo(i + j,
3));
        #undef foo
        printf("%d\n",foo(i + j,
3));
    }
    int foo(int x, int y)
    {
        return x / y + x;
    }
```

a) -8 -4
b) Compile time error
c) -8 -8
d) Undefined behaviour

ANSWER:- a
Explanation: None.

9. What is the advantage of #define over const?
a) Data type is flexible
b) Can have a pointer
c) Reduction in the size of the program
d) None of the mentioned

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
```

```
4.          #define max 37;
5.          printf("%d", max);
6.      }
```

a) 37
b) Compile time error
c) Varies
d) Depends on compiler

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          #define max 37
5.          printf("%d", max);
6.      }
```

a) 37
b) Run time error
c) Varies
d) Depends on compiler

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          #define const int
5.          const max = 32;
6.          printf("%d", max);
7.      }
```

a) Run time error
b) 32
c) int
d) const

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          #define max 45
5.          max = 32;
6.          printf("%d", max);
7.      }
```

a) 32
b) 45
c) Compile time error
d) Varies

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      # define max
3.      void m()
4.      {
5.          printf("hi");
6.      }
7.      void main()
8.      {
9.          max;
10.          m();
11.      }
```

a) Run time error
b) hi hi
c) Nothing
d) hi

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #define A 1 + 2
3.      #define B 3 + 4
4.      int main()
5.      {
6.          int var = A * B;
7.          printf("%d\n", var);
8.      }
```

a) 9
b) 11
c) 12
d) 21

ANSWER:- b
Explanation: None.

7. Which of the following Macro substitution are accepted in C?
a)

```
#define A #define
A VAR 20
```

b)

```
#define A define
#A VAR 20
```

c)

```
#define #A #define
#A VAR 20
```

d) None of the mentioned

ANSWER:- d
Explanation: None.

8. Comment on the output of the following C code.

```
#include <stdio.h>
#define var 20);
int main()
{
    printf("%d\n", var
}
```

a) No errors, it will show the output 20
b) Compile time error, the printf braces aren't closed
c) Compile time error, there are no open braces in #define
d) None of the mentioned

ANSWER:- a
Explanation: None.

9. Which of the following properties of #define is not true?
a) You can use a pointer to #define
b) #define can be made externally available
c) They obey scope rules
d) All of the mentioned

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
#define SYSTEM 20
int main()
{
    int a = 20;
    #if SYSTEM == a
    printf("HELLO ");
    #endif
    #if SYSTEM == 20
     printf("WORLD\n");
     #endif
 }
```

a) HELLO
b) WORLD
c) HELLO WORLD
d) No Output

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
#define Cprog
int main()
{
    int a = 2;
    #ifdef Cprog
    a = 1;
    printf("%d", Cprog);
}
```

a) No output on execution
b) Output as 1
c) Output as 2
d) Compile time error

ANSWER:- d
Explanation: None.

3. The "else if" in conditional inclusion is written by?
a) #else if
b) #elseif
c) #elsif
d) #elif

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
#define COLD
int main()
{
    #ifdef COLD
    printf("COLD\t");
    #undef COLD
    #endif
    #ifdef COLD
     printf("HOT\t");
     #endif
 }
```

a) HOT
b) COLD
c) COLD HOT
d) No Output

ANSWER:- b
Explanation: None.

5. Which of the following sequences are unaccepted in C language?
a)

```
#if
#else
#endif
```

b)

```
#if
#elif
#endif
```

c)

```
#if
#if
#endif
```

d)

```
#if
#undef
#endif
```

ANSWER:- c
Explanation: None.

6. In a conditional inclusion, if the condition that comes after the if is true, then what will happen during compilation?
a) Then the code up to the following #else or #elif or #endif is compiled
b) Then the code up to the following #endif is compiled even if #else or #elif is present
c) Then the code up to the following #eliif is compiled
d) None of the mentioned

ANSWER:- a
Explanation: None.

7. Conditional inclusion can be used for _____
a) Preventing multiple declarations of a variable
b) Check for existence of a variable and doing something if it exists
c) Preventing multiple declarations of same function
d) All of the mentioned

ANSWER:- d
Explanation: None.

8. The #elif directive cannot appear after the preprocessor #else directive.
a) True
b) False

ANSWER:- a
Explanation: None.

1. For each #if, #ifdef, and #ifndef directive.
a) There are zero or more #elif directives
b) Zero or one #else directive
c) One matching #endif directive
d) All of the mentioned

ANSWER:- d
Explanation: None.

2. The #else directive is used for _____
a) Conditionally include source text if the previous #if, #ifdef, #ifndef, or #elif test fails
b) Conditionally include source text if a macro name is not defined
c) Conditionally include source text if a macro name is defined
d) Ending conditional text

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
#define MIN 0
#if MIN
#define MAX 10
#endif
int main()
{
    printf("%d %d\n", MAX, MIN);
    return 0;
}
```

a) 10 0
b) Compile time error
c) Undefined behaviour
d) None of the mentioned

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
#define MIN 0
#ifdef MIN
#define MAX 10
#endif
int main()
{
    printf("%d %d\n", MAX, MIN);
    return 0;
}
```

a) 10 0
b) Compile time error
c) Undefined behaviour
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
#define MIN 0
#if defined(MIN) + defined(MAX)
#define MAX 10
#endif
int main()
{
    printf("%d %d\n", MAX, MIN);
    return 0;
}
```

a) 10 0
b) Compile time error
c) Undefined behaviour
d) Somegarbagevalue 0

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
#define MIN 0
#if defined(MIN) - (!defined(MAX))
#define MAX 10
#endif
int main()
{
    printf("%d %d\n", MAX, MIN);
    return 0;
}
```

a) 10 0
b) Compile time error
c) Undefined behaviour
d) Somegarbagevalue 0

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
#define MIN 0
#ifdef(MIN)
#define MAX 10
#endif
int main()
```

```
7.     {
8.         printf("%d %d\n", MAX,
    MIN);
9.         return 0;
10.     }
```

a) 10 0
b) Compile time error
c) Run time error
d) Preprocessor error

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     #define MIN 0);
3.     #ifdef MIN
4.     #define MAX 10
5.     #endif
6.     int main()
7.     {
8.         printf("%d %d\n", MAX,
    MIN
9.         return 0;
10.     }
```

a) 10 0
b) Compile time error due to illegal syntax for printf
c) Undefined behaviour
d) Compile time error due to illegal MIN value

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         char *p = NULL;
5.         char *q = 0;
6.         if (p)
7.             printf(" p ");
8.         else
9.             printf("nullp");
10.         if (q)
11.             printf("q\n");
12.         else
13.             printf(" nullq\n");
14.     }
```

a) nullp nullq
b) Depends on the compiler
c) x nullq where x can be p or nullp depending on the value of NULL
d) p q

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         int i = 10;
5.         void *p = &i;
6.         printf("%d\n", (int)*p);
7.         return 0;
8.     }
```

a) Compile time error
b) Segmentation fault/runtime crash
c) 10
d) Undefined behaviour

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     int main()
3.     {
4.         int i = 10;
5.         void *p = &i;
6.         printf("%f\n",
    *(float*)p);
7.         return 0;
8.     }
```

a) Compile time error
b) Undefined behaviour
c) 10
d) 0.000000

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int *f();
int main()
{
    int *p = f();
    printf("%d\n", *p);
}
int *f()
{
     int *j =
(int*)malloc(sizeof(int));
     *j = 10;
     return j;
 }
```

a) 10
b) Compile time error
c) Segmentation fault/runtime crash since pointer to local variable is returned
d) Undefined behaviour

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int *f();
int main()
{
    int *p = f();
    printf("%d\n", *p);
}
int *f()
{
     int j = 10;
     return &j;
 }
```

a) 10
b) Compile time error
c) Segmentation fault/runtime crash
d) Undefined behaviour

ANSWER:- a
Explanation: We are returning address of a local variable which should not be done. In this specific instance, we are able to see the value of 10, which may not be the case if we call other functions before calling printf() in main().

6. Comment on the following pointer declaration.

int *ptr, p;

a) ptr is a pointer to integer, p is not
b) ptr and p, both are pointers to integer
c) ptr is a pointer to integer, p may or may not be
d) ptr and p both are not pointers to integer

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int *ptr, a = 10;
    ptr = &a;
    *ptr += 1;
    printf("%d,%d/n", *ptr,
a);
}
```

a) 10,10
b) 10,11
c) 11,10
d) 11,11

ANSWER:- d
Explanation: None.

8. Comment on the following C statement.

const int *ptr;

a) You cannot change the value pointed by ptr
b) You cannot change the pointer ptr itself
c) You May or may not change the value pointed by ptr
d) You can change the pointer as well as the value pointed by it

ANSWER:- a
Explanation: None.

1. Which is an indirection operator among the following?
a) &
b) *
c) ->

d) .

ANSWER:- b
Explanation: None.

2. Which of the following does not initialize ptr to null (assuming variable declaration of a as int a=0;)?
a) int *ptr = &a;
b) int *ptr = &a – &a;
c) int *ptr = a – a;
d) All of the mentioned

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int x = 0;
void main()
{
    int *ptr = &x;
    printf("%p\n", ptr);
    x++;
    printf("%p\n ", ptr);
}
```

a) Same address
b) Different address
c) Compile time error
d) Varies

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int x = 0;
void main()
{
    int *const ptr = &x;
    printf("%p\n", ptr);
    ptr++;
    printf("%p\n ", ptr);
}
```

a) 0 1
b) Compile time error
c) 0xbfd605e8 0xbfd605ec
d) 0xbfd605e8 0xbfd605e8

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 0;
    int *ptr = &x;
    printf("%p\n", ptr);
    ptr++;
    printf("%p\n ", ptr);
}
```

a) 0xbfd605e8 0xbfd605ec
b) 0xbfd605e8 0cbfd60520
c) 0xbfd605e8 0xbfd605e9
d) Run time error

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 0;
    int *ptr = &5;
    printf("%p\n", ptr);
}
```

a) 5
b) Address of 5
c) Nothing
d) Compile time error

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 0;
    int *ptr = &x;
    printf("%d\n", *ptr);
}
```

a) Address of x
b) Junk value
c) 0
d) Run time error

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void foo(int*);
int main()
{
    int i = 10;
    foo((&i)++);
}
void foo(int *p)
{
     printf("%d\n", *p);
 }
```

a) 10
b) Some garbage value
c) Compile time error
d) Segmentation fault/code crash

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void foo(int*);
int main()
{
    int i = 10, *p = &i;
    foo(p++);
}
void foo(int *p)
{
     printf("%d\n", *p);
 }
```

a) 10
b) Some garbage value
c) Compile time error
d) Segmentation fault

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void foo(float *);
int main()
{
    int i = 10, *p = &i;
    foo(&i);
}
void foo(float *p)
{
     printf("%f\n", *p);
 }
```

a) 10.000000
b) 0.000000
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 97, *p = &i;
    foo(&i);
    printf("%d ", *p);
}
void foo(int *p)
{
     int j = 2;
     p = &j;
     printf("%d ", *p);
 }
```

a) 2 97
b) 2 2
c) Compile time error
d) Segmentation fault/code crash

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
```

```
3.    {
4.        int i = 97, *p = &i;
5.        foo(&p);
6.        printf("%d ", *p);
7.        return 0;
8.    }
9.    void foo(int **p)
10.    {
11.        int j = 2;
12.        *p = &j;
13.        printf("%d ", **p);
14.    }
```

a) 2 2
b) 2 97
c) Undefined behaviour
d) Segmentation fault/code crash

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 11;
5.        int *p = &i;
6.        foo(&p);
7.        printf("%d ", *p);
8.    }
9.    void foo(int *const *p)
10.    {
11.        int j = 10;
12.        *p = &j;
13.        printf("%d ", **p);
14.    }
```

a) Compile time error
b) 10 10
c) Undefined behaviour
d) 10 11

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10;
5.        int *p = &i;
6.        foo(&p);
7.        printf("%d ", *p);
8.        printf("%d ", *p);
9.    }
10.    void foo(int **const p)
11.    {
12.        int j = 11;
13.        *p = &j;
14.        printf("%d ", **p);
15.    }
```

a) 11 11 11
b) 11 11 Undefined-value
c) Compile time error
d) Segmentation fault/code-crash

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10;
5.        int *const p = &i;
6.        foo(&p);
7.        printf("%d\n", *p);
8.    }
9.    void foo(int **p)
10.    {
11.        int j = 11;
12.        *p = &j;
13.        printf("%d\n", **p);
14.    }
```

a) 11 11
b) Undefined behaviour
c) Compile time error
d) Segmentation fault/code-crash

ANSWER:- a
Explanation: None.

9. Which of the following is the correct syntax to send an array as a parameter to function?
a) func(&array);
b) func(#array);

c) func(*array);
d) func(array[size]);

ANSWER:- a
Explanation: None.

1. Which of the following can never be sent by call-by-value?
a) Variable
b) Array
c) Structures
d) Both Array and Structures

ANSWER:- b
Explanation: None.

2. Which type of variables can have the same name in a different function?
a) Global variables
b) Static variables
c) Function arguments
d) Both static variables and Function arguments

ANSWER:- d
Explanation: None.

3. Arguments that take input by user before running a program are called?
a) Main function arguments
b) Main arguments
c) Command-Line arguments
d) Parameterized arguments

ANSWER:- c
Explanation: None.

4. What is the maximum number of arguments that can be passed in a single function?
a) 127
b) 253
c) 361
d) No limits in number of arguments

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m(int *p, int *q)
3.    {
4.        int temp = *p; *p = *q;
    *q = temp;
5.    }
6.    void main()
7.    {
8.        int a = 6, b = 5;
9.        m(&a, &b);
10.        printf("%d %d\n", a, b);
11.    }
```

a) 5 6
b) 6 5
c) 5 5
d) 6 6

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m(int *p)
3.    {
4.        int i = 0;
5.        for(i = 0;i < 5; i++)
6.        printf("%d\t", p[i]);
7.    }
8.    void main()
9.    {
10.        int a[5] = {6, 5, 3};
11.        m(&a);
12.    }
```

a) 0 0 0 0 0
b) 6 5 3 0 0
c) Run time error
d) 6 5 3 junk junk

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m(int p, int q)
3.    {
4.        int temp = p;
5.        p = q;
6.        q = temp;
7.    }
8.    void main()
9.    {
10.        int a = 6, b = 5;
11.        m(a, b);
12.        printf("%d %d\n", a, b);
```

```
13.     }
```

a) 5 6
b) 5 5
c) 6 5
d) 6 6

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m(int p, int q)
3.    {
4.        printf("%d %d\n", p, q);
5.    }
6.    void main()
7.    {
8.        int a = 6, b = 5;
9.        m(a);
10.    }
```

a) 6
b) 6 5
c) 6 junk value
d) Compile time error

ANSWER:- d
Explanation: None.

9. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void m(int p)
3.    {
4.        printf("%d\n", p);
5.    }
6.    void main()
7.    {
8.        int a = 6, b = 5;
9.        m(a, b);
10.        printf("%d %d\n", a, b);
11.    }
```

a) 6
b) 6 5
c) 6 junk value
d) Compile time error

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        int a[3] = {1, 2, 3};
5.        int *p = a;
6.        printf("%p\t%p", p, a);
7.    }
```

a) Same address is printed
b) Different address is printed
c) Compile time error
d) Nothing

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char *s = "hello";
5.        char *p = s;
6.        printf("%p\t%p", p, s);
7.    }
```

a) Different address is printed
b) Same address is printed
c) Run time error
d) Nothing

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char *s= "hello";
5.        char *p = s;
6.        printf("%c\t%c", p[0],
    s[1]);
7.    }
```

a) Run time error
b) h h
c) h e
d) h l

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s= "hello";
    char *p = s;
    printf("%c\t%c", *(p + 3),  s[1]);
}
```

a) h e
b) l l
c) l o
d) l e

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s= "hello";
    char *p = s;
    printf("%c\t%c", 1[p], s[1]);
}
```

a) h h
b) Run time error
c) l l
d) e e

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void foo( int[] );
int main()
{
    int ary[4] = {1, 2, 3, 4};
    foo(ary);
    printf("%d ", ary[0]);
}
void foo(int p[4])
 {
     int i = 10;
     p = &i;
     printf("%d ", p[0]);
 }
```

a) 10 10
b) Compile time error
c) 10 1
d) Undefined behaviour

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[4] = {1, 2, 3, 4};
    int *p = ary + 3;
    printf("%d\n", p[-2]);
}
```

a) 1
b) 2
c) Compile time error
d) Some garbage value

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[4] = {1, 2, 3, 4};
    int *p = ary + 3;
    printf("%d %d\n", p[-2], ary[*p]);
}
```

a) 2 3
b) Compile time error
c) 2 4
d) 2 somegarbagevalue

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[4] = {1, 2, 3, 4};
    printf("%d\n", *ary);
}
```

a) 1
b) Compile time error
c) Some garbage value
d) Undefined variable

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    const int ary[4] = {1, 2, 3, 4};
    int *p;
    p = ary + 3;
    *p = 5;
    printf("%d\n", ary[3]);
}
```

a) 4
b) 5
c) Compile time error
d) 3

ANSWER:- b
Explanation: None.

3. What are the different ways to initialize an array with all elements as zero?
a) int array[5] = {};
b) int array[5] = {0};
c)

```
int a = 0,  b = 0,  c = 0;
int array[5] = {a, b, c};
```

d) All of the mentioned

ANSWER:- d
Explanation: None.

4. What are the elements present in the array of the following C code?

```
int array[5] = {5};
```

a) 5, 5, 5, 5, 5
b) 5, 0, 0, 0, 0
c) 5, (garbage), (garbage), (garbage), (garbage)
d) (garbage), (garbage), (garbage), (garbage), 5

ANSWER:- b
Explanation: None.

5. Which of the following declaration is illegal?
a)

```
int a = 0, b = 1, c = 2;
int array[3] = {a, b, c};
```

b)

```
int size = 3;
int array[size];
```

c)

```
int size = 3;
int array[size] = {1, 2, 3};
```

d) All of the mentioned

ANSWER:- c
Explanation: None.

6. An array of similar data types which themselves are a collection of dissimilar data type are _____
a) Linked Lists
b) Trees
c) Array of Structure
d) All of the mentioned

ANSWER:- c
Explanation: None.

7. Comment on an array of the void data type.
a) It can store any data-type
b) It only stores element of similar data type to first element
c) It acquires the data type with the highest precision in it
d) You cannot have an array of void data type

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[4] = {1, 2, 3, 4};
    int p[4];
    p = ary;
    printf("%d\n", p[1]);
}
```

a) 1
b) Compile time error
c) Undefined behaviour
d) 2

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    double *ptr = (double *)100;
    ptr = ptr + 2;
    printf("%u", ptr);
}
```

a) 102
b) 104
c) 108
d) 116

ANSWER:- d
Explanation:None

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int *p = (int *)2;
    int *q = (int *)3;
    printf("%d", p + q);
}
```

a) 2
b) 3
c) 5
d) Compile time error

ANSWER:- d
Explanation: None.

3. Which of the following arithmetic operation can be applied to pointers a and b?
(Assuming initialization as int *a = (int *)2; int *b = (int *)3;)
a) a + b
b) a – b
c) a * b
d) a / b

ANSWER:- b
Explanation: None.

4. What is the size of *ptr in a 32-bit machine
(Assuming initialization as int *ptr = 10;)?
a) 1
b) 2
c) 4
d) 8

ANSWER:- c
Explanation: None.

5. Which of following logical operation can be applied to pointers?
(Assuming initialization int *a = 2; int *b = 3;)
a) a | b
b) a ^ b
c) a & b
d) None of the mentioned

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s = "hello";
    char *p = s;
    printf("%c\t%c", *(p + 1), s[1]);
}
```

a) h e
b) e l
c) h h

d) e e

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s = "hello";
    char *p = s;
    printf("%c\t%c", *p, s[1]);
}
```

a) e h
b) Compile time error
c) h h
d) h e

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s = "hello";
    char *n = "cjn";
    char *p = s + n;
    printf("%c\t%c", *p, s[1]);
}
```

a) h e
b) Compile time error
c) c o
d) h n

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s = "hello";
    char *p = s * 3;
    printf("%c\t%c", *p, s[1]);
}
```

a) h e
b) l e
c) Compile time error
d) l h

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *s= "hello";
    char *p = s + 2;
    printf("%c\t%c", *p, s[1]);
}
```

a) l e
b) h e
c) l l
d) h l

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    void *p;
    int a[4] = {1, 2, 3, 8};
    p = &a[3];
    int *ptr = &a[2];
    int n = p - ptr;
    printf("%d\n", n);
 }
```

a) 1
b) Compile time error
c) Segmentation fault
d) 4
ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    void *p;
    int a[4] = {1, 2, 3, 4};
    p = &a[3];
    int *ptr = &a[2];
    int n = (int*)p - ptr;
    printf("%d\n", n);
 }
```

a) 1
b) Compile time error
c) Segmentation fault
d) 4

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int a[4] = {1, 2, 3, 4};
    int b[4] = {1, 2, 3, 4};
    int n = &b[3] - &a[2];
    printf("%d\n", n);
}
```

a) -3
b) 5
c) 4
d) Compile time error

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int a[4] = {1, 2, 3, 4};
    int *p = &a[1];
    int *ptr = &a[2];
    ptr = ptr * 1;
    printf("%d\n", *ptr);
}
```

a) 2
b) 1
c) Compile time error
d) Undefined behaviour

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int a[4] = {1, 2, 3, 4};
    int *ptr  =  &a[2];
    float n = 1;
    ptr = ptr + n;
    printf("%d\n", *ptr);
}
```

a) 4
b) 3
c) Compile time error
d) Undefined behaviour

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int a[4] = {1, 2, 3, 4};
    void *p = &a[1];
    void *ptr = &a[2];
    int n = 1;
    n = ptr - p;
    printf("%d\n", n);
 }
```

a) 1
b) 4
c) Compile time error
d) Depends on the compiler

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```c
#include <stdio.h>
```

```
2.    int main()
3.    {
4.        char *str = "hello,
    world\n";
5.        char *strc = "good
    morning\n";
6.        strcpy(strc, str);
7.        printf("%s\n", strc);
8.        return 0;
9.    }
```

a) hello, world
b) Crash/segmentation fault
c) Undefined behaviour
d) Run time error

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello
    world";
5.        char strc[] = "good
    morning india\n";
6.        strcpy(strc, str);
7.        printf("%s\n", strc);
8.        return 0;
9.    }
```

a) hello world
b) hello worldg india
c) Compile time error
d) Undefined behaviour

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello,
    world!!\n";
5.        char strc[] = "good
    morning\n";
6.        strcpy(strc, str);
7.        printf("%s\n", strc);
8.        return 0;
9.    }
```

a) hello, world!!
b) Compile time error
c) Undefined behaviour
d) Segmenation fault

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello,
    world\n";
5.        str[5] = '.';
6.        printf("%s\n", str);
7.        return 0;
8.    }
```

a) hello. world
b) hello, world
c) Compile error
d) Segmentation fault

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char str[] = "hello,
    world";
5.        str[5] = '.';
6.        printf("%s\n", str);
7.        return 0;
8.    }
```

a) hello. world
b) hello, world
c) Compile error
d) Segmentation fault

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *str = "hello world";
    char strary[] = "hello world";
    printf("%d %d\n", sizeof(str), sizeof(strary));
    return 0;
}
```

a) 11 11
b) 12 12
c) 4 12
d) 4 11

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *str = "hello world";
    char strary[] = "hello world";
    printf("%d %d\n", strlen(str), strlen(strary));
    return 0;
}
```

a) 11 11
b) 12 11
c) 11 12
d) x 11 where x can be any positive integer.

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void f(char *k)
{
    k++;
    k[2] = 'm';
    printf("%c\n", *k);
}
void main()
{
    char s[] = "hello";
    f(s);
}
```

a) l
b) e
c) h
d) o

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
void fun(char *k)
{
    printf("%s", k);
}
void main()
{
    char s[] = "hello";
    fun(s);
}
```

a) hello
b) Run time error
c) Nothing
d) h

ANSWER:- a
Explanation: None.

1. Comment on the output of the following C code.

```
#include <stdio.h>
int main()
{
    char *str = "This" //Line 1
    char *ptr = "Program\n"; //Line 2
    str = ptr; //Line 3
    printf("%s, %s\n", str, ptr); //Line 4
```

```
8.      }
```

a) Memory holding “this” is cleared at line 3
b) Memory holding “this” loses its reference at line 3
c) You cannot assign pointer like in Line 3
d) Output will be This, Program

ANSWER:- b
Explanation: None.

2. What type of initialization is needed for the segment “ptr[3] = ‘3’;” to work?
a) char *ptr = “Hello!”;
b) char ptr[] = “Hello!”;
c) both char *ptr = “Hello!”; and char ptr[] = “Hello!”;
d) none of the mentioned

ANSWER:- b
Explanation: None.

3. What is the syntax for constant pointer to address (i.e., fixed pointer address)?
a) const <type> * <name>
b) <type> * const <name>
c) <type> const * <name>
d) none of the mentioned

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int add(int a, int b)
3.    {
4.        return a + b;
5.    }
6.    int main()
7.    {
8.        int (*fn_ptr)(int, int);
9.        fn_ptr = add;
10.        printf("The sum of two
   numbers is: %d", (int)fn_ptr(2,
   3));
11.    }
```

a) Compile time error, declaration of a function inside main
b) Compile time error, no definition of function fn_ptr
c) Compile time error, illegal application of statement fn_ptr = add
d) No Run time error, output is 5

ANSWER:- d
Explanation: None.

5. What is the correct way to declare and assign a function pointer?

(Assuming the function to be assigned is "int multi(int, int);")

a) int (*fn_ptr)(int, int) = multi;
b) int *fn_ptr(int, int) = multi;
c) int *fn_ptr(int, int) = &multi;
d) none of the mentioned

ANSWER:- a
Explanation: None.

6. Calling a function f with a an array variable a[3] where a is an array, is equivalent to _____
a) f(a[3])
b) f(*(a + 3))
c) f(3[a])
d) all of the mentioned

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f(char *k)
3.    {
4.        k++;
5.        k[2] = 'm';
6.    }
7.    void main()
8.    {
9.        char s[] = "hello";
10.        f(s);
11.        printf("%c\n", *s);
12.    }
```

a) h
b) e
c) m
d) o;

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char s[] = "hello";
    s++;
    printf("%c\n", *s);
}
```

a) Compile time error
b) h
c) e
d) o

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    printf("%d%d%d\n", k, *p, **m);
}
```

a) 5 5 5
b) 5 5 junk value
c) 5 junk junk
d) Run time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    printf("%d%d%d\n", k, *p, **p);
}
```

a) 5 5 5
b) 5 5 junk value
c) 5 junk junk
d) Compile time error

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    **m = 6;
    printf("%d\n", k);
}
```

a) 5
b) Compile time error
c) 6
d) Junk

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a[3] = {1, 2, 3};
    int *p = a;
    int *r = &p;
    printf("%d", (**r));
}
```

a) 1
b) Compile time error
c) Address of a
d) Junk value

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a[3] = {1, 2, 3};
    int *p = a;
    int **r = &p;
```

```
7.          printf("%p %p", *r, a);
8.      }
```

a) Different address is printed
b) 1 2
c) Same address is printed
d) 1 1

ANSWER:- c
Explanation: None.

6. How many number of pointer (*) does C have against a pointer variable declaration?
a) 7
b) 127
c) 255
d) No limits

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int a = 1, b = 2, c = 3;
5.          int *ptr1 = &a, *ptr2 =
    &b, *ptr3 = &c;
6.          int **sptr = &ptr1; //-
    Ref
7.          *sptr = ptr2;
8.      }
```

a) ptr1 points to a
b) ptr1 points to b
c) sptr points to ptr2
d) none of the mentioned

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          int a[3] = {1, 2, 3};
5.          int *p = a;
6.          int **r = &p;
7.          printf("%p %p", *r, a);
8.      }
```

a) Different address is printed
b) 1 2
c) Same address is printed
d) 1 1

ANSWER:- c
Explanation: None.

1. What substitution should be made to //-Ref such that ptr1 points to variable c in the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int a = 1, b = 2, c = 3;
5.          int *ptr1 = &a;
6.          int **sptr = &ptr1;
7.          //-Ref
8.      }
```

a) *sptr = &c;
b) **sptr = &c;
c) *ptr1 = &c;
d) none of the mentioned

ANSWER:- a
Explanation: None.

2. Which of the following declaration will result in run-time error?
a) int **c = &c;
b) int **c = &*c;
c) int **c = **c;
d) none of the mentioned

ANSWER:- d
Explanation: None.

3. Comment on the output of the following C code.

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          int a = 10;
5.          int **c -= &&a;
6.      }
```

a) You cannot apply any arithmetic operand to a pointer
b) We don't have address of an address operator
c) We have address of an address operator

d) None of the mentioned

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    printf("%d%d%d\n", k, *p, **m);
}
```

a) 5 5 5
b) 5 5 junk value
c) 5 junk junk
d) Compile time error

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    printf("%d%d%d\n", k, *p, **p);
}
```

a) 5 5 5
b) 5 5 junk value
c) 5 junk junk
d) Compile time error

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    **m = 6;
    printf("%d\n", k);
}
```

a) 5
b) Run time error
c) 6
d) Junk

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a[3] = {1, 2, 3};
    int *p = a;
    int *r = &p;
    printf("%d", (**r));
}
```

a) 1
b) Compile time error
c) Address of a
d) Junk value

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a[2][3] = {1, 2, 3, 4, 5};
    int i = 0, j = 0;
    for (i = 0; i < 2; i++)
    for (j = 0; j < 3; j++)
    printf("%d", a[i][j]);
}
```

a) 1 2 3 4 5 0
b) 1 2 3 4 5 junk
c) 1 2 3 4 5 5
d) Run time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int a[2][3] = {1, 2, 3, ,
4, 5};
    int i = 0, j = 0;
    for (i = 0; i < 2; i++)
    for (j = 0; j < 3; j++)
    printf("%d", a[i][j]);
}
```

a) 1 2 3 junk 4 5
b) Compile time error
c) 1 2 3 0 4 5
d) 1 2 3 3 4 5

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void f(int a[][3])
{
    a[0][1] = 3;
    int i = 0, j = 0;
    for (i = 0; i < 2; i++)
    for (j = 0; j < 3; j++)
    printf("%d", a[i][j]);
}
 void main()
 {
     int a[2][3] = {0};
     f(a);
 }
```

a) 0 3 0 0 0 0
b) Junk 3 junk junk junk junk
c) Compile time error
d) All junk values

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void f(int a[][])
{
    a[0][1] = 3;
    int i = 0, j = 0;
    for (i = 0;i < 2; i++)
    for (j = 0;j < 3; j++)
    printf("%d", a[i][j]);
}
 void main()
 {
     int a[2][3] = {0};
     f(a);
 }
```

a) 0 3 0 0 0 0
b) Junk 3 junk junk junk junk
c) Compile time error
d) All junk values

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void f(int a[2][])
{
    a[0][1] = 3;
    int i = 0, j = 0;
    for (i = 0;i < 2; i++)
    for (j = 0;j < 3; j++)
    printf("%d", a[i][j]);
}
 void main()
 {
     int a[2][3] = {0};
     f(a);
 }
```

a) 0 3 0 0 0 0
b) Junk 3 junk junk junk junk
c) Compile time error
d) All junk values

ANSWER:- c
Explanation: None.

6. Comment on the following C statement.

int (*a)[7];

a) An array “a” of pointers
b) A pointer “a” to an array
c) A ragged array
d) None of the mentioned

ANSWER:- b
Explanation: None.

7. Comment on the following 2 arrays with respect to P and Q.

```
int *a1[8];
int *(a2[8]);
P. Array of pointers
Q. Pointer to an array
```

a) a1 is P, a2 is Q
b) a1 is P, a2 is P
c) a1 is Q, a2 is P
d) a1 is Q, a2 is Q

ANSWER:- b
Explanation: None.

8. Which of the following is not possible statically in C?
a) Jagged Array
b) Rectangular Array
c) Cuboidal Array
d) Multidimensional Array

ANSWER:- a
Explanation: None.

1. What is the correct syntax to send a 3-dimensional array as a parameter? (Assuming declaration int a[5][4][3];)
a) func(a);
b) func(&a);
c) func(*a);
d) func(**a);

ANSWER:- a
Explanation: None.

2. What are the applications of a multidimensional array?
a) Matrix-Multiplication
b) Minimum Spanning Tree
c) Finding connectivity between nodes
d) All of the mentioned

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
void foo(int *ary[]);
int main()
{
    int ary[2][3];
    foo(ary);
}
void foo(int *ary[])
{
     int i = 10, j = 2, k;
     ary[0] = &i;
     ary[1] = &j;
     *ary[0] = 2;
     for (k = 0;k < 2; k++)
     printf("%d\n", *ary[k]);
 }
```

a) 2 2
b) Compile time error
c) Undefined behaviour
d) 10 2

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void foo(int (*ary)[3]);
int main()
{
    int ary[2][3];
    foo(ary);
}
void foo(int (*ary)[3])
{
     int i = 10, j = 2, k;
     ary[0] = &i;
     ary[1] = &j;
     for (k = 0;k < 2; k++)
     printf("%d\n", *ary[k]);
 }
```

a) Compile time error
b) 10 2
c) Undefined behaviour
d) segmentation fault/code crash

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    foo(ary);
}
void foo(int **ary)
{
    int i = 10, k = 20, j = 30;
    int *ary[2];
     ary[0] = &i;
     ary[1] = &j;
     printf("%d\n", ary[0][1]);
 }
```

a) 10
b) 20
c) Compile time error
d) Undefined behaviour

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[2][3][4], j = 20;
    ary[0][0] = &j;
    printf("%d\n", *ary[0][0]);
}
```

a) Compile time error
b) 20
c) Address of j
d) Undefined behaviour

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int ary[2][3];
    ary[][] = {{1, 2, 3}, {4, 5, 6}};
    printf("%d\n", ary[1][0]);
 }
```

a) Compile time error
b) 4
c) 1
d) 2

ANSWER:- a
Explanation: None.

1. Which of the following is the correct syntax to declare a 3 dimensional array using pointers?
a) char *a[][];
b) char **a[];
c) char ***a;
d) all of the mentioned

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *a = {"p", "r", "o", "g", "r", "a", "m"};
    printf("%s", a);
}
```

a) Output will be program
b) Output will be p
c) No output
d) Compile-time error

ANSWER:- b
Explanation: None.

3. An array of strings can be initialized by _____
a) char *a[] = {“Hello”, “World”};
b) char *a[] = {“Hello”, “Worlds”};
c)

```
char *b = "Hello";

char *c = "World";

char *a[] = {b, c};
```

d) all of the mentioned

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *a[10] = {"hi", "hello", "how"};
    int i = 0;
    for (i = 0;i < 10; i++)
    printf("%s", *(a[i]));
}
```

a) segmentation fault
b) hi hello how followed by 7 null values
c) 10 null values
d) depends on compiler

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *a[10] = {"hi", "hello", "how"};
    int i = 0, j = 0;
    a[0] = "hey";
    for (i = 0;i < 10; i++)
    printf("%s\n", a[i]);
}
```

a) hi hello how Segmentation fault
b) hi hello how followed by 7 null values
c) hey hello how Segmentation fault
d) depends on compiler

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code on a 32-bit system?

```
#include <stdio.h>
void main()
{
    char *a[10] = {"hi", "hello", "how"};
    printf("%d\n", sizeof(a));
}
```

a) 10
b) 13
c) Run time error
d) 40

ANSWER:- d
Explanation: If the system is 32-bit system, then the size of pointer will be 4 bytes. For such a system, the size of array a will be 4×10 = 40 bytes. The size of pointer is 8 bytes on a 64 bit system. For the given array a of 10 elements, it will be 8×10 = 80 bytes.

7. What will be the output of the following C code on a 32-bit system?

```
#include <stdio.h>
void main()
{
    char *a[10] = {"hi", "hello", "how"};
    printf("%d\n", sizeof(a[1]));
}
```

a) 6
b) 4
c) 5
d) 3

ANSWER:- b
Explanation: Array element a[1] is storing an address of character pointer. For a 32-bit systems its 4 bytes and for a 64-bit system, its 8 bytes.

8. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    char *a[10] = {"hi", "hello", "how"};
    int i = 0;
    for (i = 0;i < 10; i++)
    printf("%s", a[i]);
}
```

a) hi hello how Segmentation fault
b) hi hello how null
c) hey hello how Segmentation fault
d) hi hello how followed by 7 nulls

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *p[1] = {"hello"};
    printf("%s", (p)[0]);
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) hello
d) None of the mentioned

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char **p = {"hello", "hi", "bye"};
    printf("%s", (p)[0]);
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) hello
d) Address of hello

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 1;
    int *a[] = {&i, &j};
    printf("%d", (*a)[0]);
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) 0
d) Some garbage value

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 1;
    int *a[] = {&i, &j};
    printf("%d", *a[0]);
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) 0
d) Some garbage value

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 0, j = 1;
    int *a[] = {&i, &j};
    printf("%d", (*a)[1]);
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) 1
d) Some garbage value

ANSWER:- d
Explanation: None.

6. Which of the following are generated from char pointer?

a) char *string = “Hello.”;
b)

```
char *string;

scanf("%s", string);
```

c) char string[] = “Hello.”;
d) char *string = “Hello.”; and char string[] = “Hello.”;

ANSWER:- a
Explanation: None.

7. Which of the following declaration are illegal?
a) int a[][] = {{1, 2, 3}, {2, 3, 4, 5}};
b) int *a[] = {{1, 2, 3}, {2, 3, 4, 5}};
c) int a[4][4] = {{1, 2, 3}, {2, 3, 4, 5}};
d) none of the mentioned

ANSWER:- a
Explanation: None.

1. Which is true for a, if a is defined as “int a[10][20];”?
a) a is true two-dimensional array
b) 200 int-sized locations have been set aside
c) The conventional rectangular subscript calculation 20 * row + col is used to find the element a[row, col].
d) All of the mentioned

ANSWER:- d
Explanation: None.

2. Which is true for b, if b is defined as “int *b[10];”?
a) The definition only allocates 10 pointers and does not initialize them
b) Initialization must be done explicitly
c) The definition only allocates 10 pointers and does not initialize them & Initialization must be done explicitly
d) Error

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char a[10][5] = {"hi",
   "hello", "fellows"};
5.        printf("%s", a[2]);
6.    }
```

a) fellows
b) fellow
c) fello
d) fell

ANSWER:- c
Explanation: Since every row in the array a[10][5] can contain only 5 characters, the a[2] element will hold “fello” i.e. 5 characters. There will not be any null character in a[2]. Since, the array is completely intialized, other rows (row a[3] will have only null characters. Hence, printf() using %s specifier will display fello only.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char a[10][5] = {"hi",
   "hello", "fellows"};
5.        printf("%p\n", a);
6.        printf("%p", a[0]);
7.    }
```

a) same address is printed
b) different address is printed
c) hello
d) hi hello fello

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char a[10][5] = {"hi",
   "hello", "fellows"};
5.        printf("%d",
   sizeof(a[1]));
6.    }
```

a) 2
b) 4
c) 5
d) 10

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char a[1][5] = {"hello"};
    printf("%s", a[0]);
    return 0;
}
```

a) Compile time error
b) hello
c) Undefined behaviour
d) hellon

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *a[1] = {"hello"};
    printf("%s", a[0]);
    return 0;
}
```

a) Compile time error
b) hello
c) Undefined behaviour
d) hellon

ANSWER:- b
Explanation: None.

8. Which of the following statements are true?

P. Pointer to Array

Q. Multi-dimensional array

a) P are static, Q are static
b) P are static, Q are dynamic
c) P are dynamic, Q are static
d) P are dynamic, Q are dynamic

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code (considering sizeof char is 1 and pointer is 4)?

```
#include <stdio.h>
int main()
{
    char *a[2] = {"hello", "hi"};
    printf("%d", sizeof(a));
    return 0;
}
```

a) 9
b) 4
c) 8
d) 10

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char a[2][6] = {"hello", "hi"};
    printf("%d", sizeof(a));
    return 0;
}
```

a) 9
b) 12
c) 8
d) 10

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char a[2][6] = {"hello", "hi"};
    printf("%s", *a + 1);
    return 0;
}
```

a) hello
b) hi

c) ello
d) ello hi

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *a[2] = {"hello",
    "hi"};
5.        printf("%s", *(a + 1));
6.        return 0;
7.    }
```

a) hello
b) ello
c) hi
d) ello hi

ANSWER:- c
Explanation: None.

5. What is the advantage of a multidimensional array over pointer array?
a) Predefined size
b) Input can be taken from user
c) Faster Access
d) All of the mentioned

ANSWER:- d
Explanation: None.

6. Which of the following operation is possible using a pointer char? (Assuming the declaration is char *a;)
a) Input via %s
b) Generation of the multidimensional array
c) Changing address to point at another location
d) All of the mentioned

ANSWER:- c
Explanation: None.

7. Comment on the following two operations.

```
    int *a[] = {{1, 2, 3}, {1, 2, 3, 4}};
//- 1
    int b[4][4] = {{1, 2, 3}, {1, 2, 3,
4}};//- 2
```

a) 1 will work, 2 will not
b) 1 and 2, both will work
c) 1 won't work, 2 will work
d) Neither of them will work

ANSWER:- c
Explanation: None.

8. Comment on the following two operations.

```
    int *a[] = {{1, 2, 3}, {1, 2, 3, 4}};
//- 1
    int b[][] = {{1, 2, 3}, {1, 2, 3, 4}};
//- 2
```

a) 1 works, 2 doesn't
b) 2 works, 1 doesn't
c) Both of them work
d) Neither of them work

ANSWER:- d
Explanation: None.

1. What does argc and argv indicate in command-line arguments?
(Assuming: int main(int argc, char *argv[]))
a) argument count, argument variable
b) argument count, argument vector
c) argument control, argument variable
d) argument control, argument vector

ANSWER:- b
Explanation: None.

2. Which of the following syntax is correct for command-line arguments?
a)

```
int main(int var, char *varg[])
```

b)

```
int main(char *argv[], int argc)
```

c)

```
    int main()
    {
        int argv, char *argc[];
    }
```

d) none of the mentioned

ANSWER:- a
Explanation: None.

3. In linux, argv[0] by command-line argument can be occupied by _____
a) ./a.out
b) ./test
c) ./fun.out.out
d) all of the mentioned

ANSWER:- d
Explanation: All the options mentioned (./a.out, ./test, ./fun.out.out) are simply the command without any argument. A command is always stored as argument vector zero i.e., argv[0] always contain the command where as argv[1], argv[2], etc. contains the arguments to the commands, if any.

4. What type of array is generally generated in Command-line argument?
a) Single dimension array
b) 2-Dimensional Square Array
c) Jagged Array
d) 2-Dimensional Rectangular Array

ANSWER:- c
Explanation: None.

5. What will be the output of the following C statement? (assuming the input is “cool brother in city”)

```
printf(“%s\n”, argv[argc]);
```

a) (null)
b) City
c) In
d) Segmentation Fault

ANSWER:- a
Explanation: None.

6. What is the first argument in command line arguments?
a) The number of command-line arguments the program was invoked with;
b) A pointer to an array of character strings that contain the arguments
c) Nothing
d) None of the mentioned

ANSWER:- a
Explanation: None.

7. What is the second argument in command line arguments?
a) The number of command-line arguments the program was invoked with;
b) A pointer to an array of character strings that contain the arguments, one per string
c) Nothing
d) None of the mentioned

ANSWER:- b
Explanation: None.

8. What is argv[0] in command line arguments?
a) The name by which the program was invoked
b) The name of the files which are passed to the program
c) Count of the arguments in argv[] vector
d) None of the mentioned

ANSWER:- a
Explanation: None.

1. A program that has no command line arguments will have argc _____
a) Zero
b) Negative
c) One
d) Two

ANSWER:- c
Explanation: None.

2. What is the index of the last argument in command line arguments?
a) argc – 2
b) argc + 1
c) argc
d) argc – 1

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code (if run with no options or arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        printf("%d\n", argc);
5.        return 0;
6.    }
```

a) 0
b) 1
c) Depends on the platform
d) Depends on the compiler

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code (run without any command line arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        while (argc--)
5.        printf("%s\n",
   argv[argc]);
6.        return 0;
7.    }
```

a) Compile time error
b) Executablefilename
c) Segmentation fault
d) Undefined

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code (run without any command line arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        printf("%s\n",
   argv[argc]);
5.        return 0;
6.    }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code (run without any command line arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        while (*argv++ != NULL)
5.        printf("%s\n", *argv);
6.        return 0;
7.    }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code (run without any command line arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        while (*argv  !=  NULL)
5.        printf("%s\n",
   *(argv++));
6.        return 0;
7.    }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code (run without any command line arguments)?

```
1.    #include <stdio.h>
2.    int main(int argc, char
   *argv[])
3.    {
4.        while (argv != NULL)
5.        printf("%s\n",
   *(argv++));
6.        return 0;
7.    }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

ANSWER:- a
Explanation: None.

1. Which function is not called in the following C program?

```
1.    #include <stdio.h>
```

```
2.    void first()
3.    {
4.        printf("first");
5.    }
6.    void second()
7.    {
8.        first();
9.    }
10.    void third()
11.    {
12.        second();
13.    }
14.    void main()
15.    {
16.        void (*ptr)();
17.        ptr = third;
18.        ptr();
19.    }
```

a) Function first
b) Function second
c) Function third
d) None of the mentioned

ANSWER:- d
Explanation: None.

2. How to call a function without using the function name to send parameters?
a) typedefs
b) Function pointer
c) Both typedefs and Function pointer
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. Which of the following is a correct syntax to pass a Function Pointer as an argument?
a) void pass(int (*fptr)(int, float, char)){}
b) void pass(*fptr(int, float, char)){}
c) void pass(int (*fptr)){}
d) void pass(*fptr){}

ANSWER:- a
Explanation: None.

4. Which of the following is not possible in C?
a) Array of function pointer
b) Returning a function pointer
c) Comparison of function pointer
d) None of the mentioned

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void first()
3.    {
4.        printf("Hello World");
5.    }
6.    void main()
7.    {
8.        void *ptr() = first;
9.        ptr++
10.        ptr();
11.    }
```

a) Illegal application of ++ to void data type
b) pointer function initialized like a variable
c) Illegal application of ++ to void data type & pointer function initialized like a variable
d) None of the mentioned

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int mul(int a, int b, int c)
3.    {
4.        return a * b * c;
5.    }
6.    void main()
7.    {
8.        int
   (*function_pointer)(int, int,
   int);
9.        function_pointer  =  mul;
10.        printf("The product of
   three numbers is:%d",
11.        function_pointer(2, 3,
   4));
12.    }
```

a) The product of three numbers is:24
b) Run time error
c) Nothing
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int mul(int a, int b, int c)
{
    return a * b * c;
}
void main()
{
    int (function_pointer)(int, int, int);
    function_pointer = mul;
     printf("The product of three numbers is:%d", function_pointer(2, 3, 4));
}
```

a) The product of three numbers is:24
b) Compile time error
c) Nothing
d) Varies

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void f(int (*x)(int));
int myfoo(int);
int (*fooptr)(int);
int ((*foo(int)))(int);
int main()
{
    fooptr = foo(0);
    fooptr(10);
 }
 int ((*foo(int i)))(int)
 {
     return myfoo;
 }
 int myfoo(int i)
 {
     printf("%d\n", i + 1);
 }
```

a) 10
b) 11
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int mul(int a, int b, int c)
{
    return a * b * c;
}
void main()
{
    int *function_pointer;
    function_pointer = mul;
     printf("The product of three numbers is:%d", function_pointer(2, 3, 4));
}
```

a) The product of three numbers is:24
b) Compile time error
c) Nothing
d) Varies

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int sub(int a, int b, int c)
{
    return a - b - c;
}
void main()
{
    int (*function_pointer)(int, int, int);
    function_pointer = &sub;
     printf("The difference of three numbers is:%d",
```

```
11.         (*function_pointer)(2,
    3, 4));
12.     }
```

a) The difference of three numbers is:1
b) Run time error
c) The difference of three numbers is:-5
d) Varies

ANSWER:- c
Explanation: None.

3. One of the uses for function pointers in C is _____
a) Nothing
b) There are no function pointers in c
c) To invoke a function
d) To call a function defined at run-time

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f(int);
3.    void (*foo)() = f;
4.    int main(int argc, char
    *argv[])
5.    {
6.        foo(10);
7.        return 0;
8.    }
9.    void f(int i)
10.    {
11.        printf("%d\n", i);
12.    }
```

a) Compile time error
b) 10
c) Undefined behaviour
d) None of the mentioned

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f(int);
3.    void (*foo)(void) = f;
4.    int main(int argc, char
    *argv[])
5.    {
6.        foo(10);
7.        return 0;
8.    }
9.    void f(int i)
10.    {
11.        printf("%d\n", i);
12.    }
```

a) Compile time error
b) 10
c) Undefined behaviour
d) None of the mentioned

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f(int);
3.    void (*foo)(float) = f;
4.    int main()
5.    {
6.        foo(10);
7.    }
8.    void f(int i)
9.    {
10.        printf("%d\n", i);
11.    }
```

a) Compile time error
b) 10
c) 10.000000
d) Undefined behaviour

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void f(int (*x)(int));
3.    int myfoo(int i);
4.    int (*foo)(int) = myfoo;
5.    int main()
6.    {
7.        f(foo(10));
8.    }
9.    void f(int (*i)(int))
```

```
10.     {
11.         i(11);
12.     }
13.     int myfoo(int i)
14.     {
15.         printf("%d\n", i);
16.         return i;
17.     }
```

a) Compile time error
b) Undefined behaviour
c) 10 11
d) 10 Segmentation fault

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     void f(int (*x)(int));
3.     int myfoo(int);
4.     int (*foo)() = myfoo;
5.     int main()
6.     {
7.         f(foo);
8.     }
9.     void f(int(*i)(int ))
10.     {
11.         i(11);
12.     }
13.     int myfoo(int i)
14.     {
15.         printf("%d\n", i);
16.         return i;
17.     }
```

a) 10 11
b) 11
c) 10
d) Undefined behaviour

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     void main()
3.     {
4.         struct student
5.         {
6.             int no;
7.             char name[20];
8.         };
9.         struct student s;
10.         no = 8;
11.         printf("%d", no);
12.     }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct student
3.     {
4.         int no;
5.         char name[20];
6.     };
7.     void main()
8.     {
9.         struct student s;
10.         s.no = 8;
11.         printf("hello");
12.     }
```

a) Run time error
b) Nothing
c) hello
d) Varies

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct student
3.     {
4.         int no = 5;
5.         char name[20];
6.     };
7.     void main()
```

```
{
    struct student s;
     s.no = 8;
     printf("hello");
 }
```

a) Nothing
b) Compile time error
c) hello
d) Varies

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    int no;
    char name[20];
};
void main()
{
    student s;
     s.name = "hello";
     printf("hello");
 }
```

a) Nothing
b) hello
c) Compile time error
d) Varies

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    struct student
    {
        int no;
        char name[20];
    };
    struct student s;
     s.no = 8;
     printf("%s", s.name);
 }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    int no;
    char name[20];
};
struct student s;
void main()
{
     s.no = 8;
     printf("%s", s.name);
 }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int *((*x)())[2];
    x();
    printf("after x\n");
}
int *((*x)())[2]
{
     int **str;
     str =
(int*)malloc(sizeof(int)* 2);
     return str;
 }
```

a) Compile time error
b) Undefined behaviour
c) After x
d) None of the mentioned

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
void (*(f)())(int, float);
void (*(*x)())(int, float) =
f;
void ((*y)(int, float));
void foo(int i, float f);
int main()
{
    y = x();
    y(1, 2);
 }
 void (*(f)())(int, float)
 {
     return foo;
 }
 void foo(int i, float f)
 {
     printf("%d %f\n", i, f);
 }
```

a) 1 2.000000
b) 1 2
c) Compile time error
d) Segmentation fault/code crash

ANSWER:- a
Explanation: None.

9. What does this declaration say?

```
int (*(*y)())[2];
```

a) y is pointer to the function which returns pointer to integer array
b) y is pointer to the function which returns array of pointers
c) y is function which returns function pointer which in turn returns pointer to integer array
d) y is function which returns array of integers

ANSWER:- a
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
void (*(f)())(int, float);
typedef void (*(*x)())(int,
float);
void foo(int i, float f);
int main()
{
    x = f;
    x();
}
 void (*(f)())(int, float)
 {
     return foo;
 }
 void foo(int i, float f)
 {
     printf("%d %f\n", i, f);
 }
```

a) Compile time error
b) Undefined behaviour
c) 1 2.000000
d) Nothing

ANSWER:- a
Explanation: None.

11. What will be the output of the following C code?

```
#include <stdio.h>
void (*(f)())(int, float);
typedef void (*(*x)())(int,
float);
void foo(int i, float f);
int main()
{
    x p = f;
    p();
}
 void (*(f)())(int, float)
 {
     return foo;
 }
 void foo(int i, float f)
 {
```

```
16.         printf("%d %f\n", i, f);
17.     }
```

a) Compile time error
b) Undefined behaviour
c) 1 2.000000
d) Nothing

ANSWER:- d
Explanation: None.

1. Read the following expression?

```
void (*ptr)(int);
```

a) ptr is pointer to int that converts its type to void
b) ptr is pointer to function passing int returning void
c) ptr is pointer to void that converts its type to int
d) ptr is pointer to function passing void returning int

ANSWER:- b
Explanation: None.

2. Which of the following expression is true for the following C statement?

```
ptr is array with 3 elements of pointer to function returning pointer of int
```

a) int **ptr[3]();
b) int *(*ptr[3])();
c) int (*(*ptr[3])());
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. What makes the following declaration denote?

```
int **ptr;
```

a) ptr is a function pointer that returns pointer to int type
b) ptr is a pointer to an int pointer
c) ptr is a pointer to pointer to type int
d) none of the mentioned

ANSWER:- b
Explanation: None.

4. What makes the following declaration denote?

```
char *str[5];
```

a) str is an array of 5 element pointer to type char
b) str is a pointer to an array of 5 elements
c) str is a function pointer of 5 elements returning char
d) none of the mentioned

ANSWER:- a
Explanation: None.

5. Comment on the following declaration.

```
int (*ptr)(); // i)
char *ptr[]; // ii)
```

a) Both i) and ii) and cannot exist due to same name
b) i) is legal, ii) is illegal
c) i) is illegal, ii) is legal
d) Both i) and ii) will work legal and flawlessly

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      int no;
5.      char name[20];
6.  }
7.  void main()
8.  {
9.      struct student s;
10.     s.no = 8;
11.     printf("hello");
12.  }
```

a) Compile time error
b) Nothing
c) hello
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
```

```
4.          int no = 5;
5.          char name[20];
6.      };
7.      void main()
8.      {
9.          struct student s;
10.          s.no = 8;
11.          printf("hello");
12.      }
```

a) Nothing
b) Compile time error
c) hello
d) Varies

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      struct student
3.      {
4.          int no;
5.          char name[20];
6.      };
7.      void main()
8.      {
9.          student s;
10.          s.no = 8;
11.          printf("hello");
12.      }
```

a) Nothing
b) hello
c) Compile time error
d) Varies

ANSWER:- c
Explanation: None.

9. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          struct student
5.          {
6.              int no;
7.              char name[20];
8.          };
9.          struct student s;
10.          s.no = 8;
11.          printf("%d", s.no);
12.      }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- d
Explanation: None.

10. Is the below declaration legal?

```
int* ((*x)())[2];
```

a) True
b) False
c) Undefined behaviour
d) Depends on the standard

ANSWER:- b
Explanation: None.

1. Which of the following are themselves a collection of different data types?
a) string
b) structures
c) char
d) all of the mentioned

ANSWER:- b
Explanation: None.

2. User-defined data type can be derived by_____
a) struct
b) enum
c) typedef
d) all of the mentioned

ANSWER:- d
Explanation: None.

3. Which operator connects the structure name to its member name?
a) –
b) <-
c) .
d) Both <- and .

ANSWER:- c
Explanation: None.

4. Which of the following cannot be a structure member?
a) Another structure
b) Function
c) Array
d) None of the mentioned

ANSWER:- b
Explanation: None.

5. Which of the following structure declaration will throw an error?
a)

```
struct temp{}s;
main(){}
```

b)

```
struct temp{};
struct temp s;
main(){}
```

c)

```
struct temp s;
struct temp{};
main(){}
```

d) None of the mentioned

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      int no;
5.      char name[20];
6.  }
7.  void main()
8.  {
9.      struct student s;
10.     s.no = 8;
11.     printf("hello");
12.  }
```

a) Compile time error
b) Nothing
c) hello
d) Varies

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      int no = 5;
5.      char name[20];
6.  };
7.  void main()
8.  {
9.      struct student s;
10.     s.no = 8;
11.     printf("hello");
12.  }
```

a) Nothing
b) Compile time error
c) hello
d) Varies

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      int no;
5.      char name[20];
6.  };
7.  void main()
8.  {
9.      student s;
10.     s.no = 8;
11.     printf("hello");
12.  }
```

a) Nothing
b) hello
c) Compile time error
d) Varies

ANSWER:- c
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    struct student
    {
        int no;
        char name[20];
    };
    struct student s;
     s.no = 8;
     printf("%d", s.no);
 }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- d
Explanation: None.

10. Can the following C code be compiled successfully?

```
#include <stdio.h>
struct p
{
    int k;
    char c;
    float f;
};
int main()
{
     struct p x = {.c = 97, .f = 3, .k = 1};
     printf("%f\n", x.f);
 }
```

a) Yes
b) No
c) Depends on the standard
d) Depends on the platform

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    struct student
    {
        int no;
        char name[20];
    };
    struct student s;
     no = 8;
     printf("%d", no);
 }
```

a) Nothing
b) Compile time error
c) Junk
d) 8

ANSWER:- b
Explanation: None.

2. How many bytes in memory taken by the following C structure?

```
#include <stdio.h>
struct test
{
    int k;
    char c;
};
```

a) Multiple of integer size
b) integer size+character size
c) Depends on the platform
d) Multiple of word size

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
struct
{
    int k;
    char c;
};
int main()
{
    struct p;
     p.k = 10;
     printf("%d\n", p.k);
 }
```

a) Compile time error
b) 10
c) Undefined behaviour
d) Segmentation fault

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct
3.    {
4.        int k;
5.        char c;
6.    } p;
7.    int p = 10;
8.    int main()
9.    {
10.        p.k = 10;
11.        printf("%d %d\n", p.k,
   p);
12.    }
```

a) Compile time error
b) 10 10
c) Depends on the standard
d) Depends on the compiler

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct p
3.    {
4.        int k;
5.        char c;
6.    };
7.    int p = 10;
8.    int main()
9.    {
10.        struct p x;
11.        x.k = 10;
12.        printf("%d %d\n", x.k,
   p);
13.    }
```

a) Compile time error
b) 10 10
c) Depends on the standard
d) Depends on the compiler

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct p
3.    {
4.        int k;
5.        char c;
6.        float f;
7.    };
8.    int p = 10;
9.    int main()
10.    {
11.        struct p x = {1, 97};
12.        printf("%f %d\n", x.f,
   p);
13.    }
```

a) Compile time error
b) 0.000000 10
c) Somegarbage value 10
d) 0 10

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code according to C99 standard?

```
1.    #include <stdio.h>
2.    struct p
3.    {
4.        int k;
5.        char c;
6.        float f;
7.    };
8.    int main()
9.    {
10.        struct p x = {.c =
   97, .f = 3, .k = 1};
11.        printf("%f\n", x.f);
12.    }
```

a) 3.000000
b) Compile time error
c) Undefined behaviour

d) 1.000000

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code according to C99 standard?

```
#include <stdio.h>
struct p
{
    int k;
    char c;
    float f;
};
int main()
{
     struct p x = {.c = 97, .k = 1, 3};
     printf("%f \n", x.f);
 }
```

a) 3.000000
b) 0.000000
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code according to C99 standard?

```
#include <stdio.h>
struct p
{
    int k;
    char c;
    float f;
};
int main()
{
     struct p x = {.c = 97};
     printf("%f\n", x.f);
 }
```

a) 0.000000
b) Somegarbagevalue
c) Compile time error
d) None of the mentioned

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *name;
};
struct student s;
struct student fun(void)
{
    s.name = "newton";
     printf("%s\n", s.name);
     s.name = "alan";
     return s;
 }
 void main()
 {
     struct student m = fun();
     printf("%s\n", m.name);
     m.name = "turing";
     printf("%s\n", s.name);
 }
```

a) newton alan alan
b) alan newton alan
c) alan alan newton
d) compile time error

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *name;
};
void main()
{
    struct student s, m;
    s.name = "st";
     m = s;
     printf("%s%s", s.name, m.name);
```

```
12.     }
```

a) Compile time error
b) Nothing
c) Junk values
d) st st

ANSWER:- d
Explanation: None.

3. Which of the following return-type cannot be used for a function in C?
a) char *
b) struct
c) void
d) none of the mentioned

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct temp
3.    {
4.        int a;
5.    } s;
6.    void func(struct temp s)
7.    {
8.        s.a = 10;
9.        printf("%d\t", s.a);
10.    }
11.    main()
12.    {
13.        func(s);
14.        printf("%d\t", s.a);
15.    }
```

a) 10 (Garbage Value)
b) 0 10
c) 10 0
d) (Garbage Value) 10

ANSWER:- c
Explanation: None.

5. Which of the following is not possible under any scenario?
a) s1 = &s2;
b) s1 = s2;
c) (*s1).number = 10;
d) None of the mentioned

ANSWER:- d
Explanation: None.

6. Which of the following operation is illegal in structures?
a) Typecasting of structure
b) Pointer to a variable of the same structure
c) Dynamic allocation of memory for structure
d) All of the mentioned

ANSWER:- a
Explanation: None.

7. Presence of code like “s.t.b = 10” indicates _____
a) Syntax Error
b) Structure
c) double data type
d) An ordinary variable name

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct student
3.    {
4.        char *name;
5.    };
6.    struct student fun(void)
7.    {
8.        struct student s;
9.        s.name = "alan";
10.        return s;
11.    }
12.    void main()
13.    {
14.        struct student m =
    fun();
15.        s.name = "turing";
16.        printf("%s", m.name);
17.    }
```

a) alan
b) Turing
c) Compile time error
d) Nothing

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct point
3.  {
4.      int x;
5.      int y;
6.  };
7.  int main()
8.  {
9.      struct point p = {1};
10.     struct point p1 = {1};
11.     if(p == p1)
12.         printf("equal\n");
13.     else
14.         printf("not
    equal\n");
15.  }
```

a) Compile time error
b) equal
c) depends on the standard
d) not equal

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct point
3.  {
4.      int x;
5.      int y;
6.  };
7.  struct notpoint
8.  {
9.      int x;
10.     int y;
11.  };
12.  struct point foo();
13.  int main()
14.  {
15.      struct point p = {1};
16.      struct notpoint p1 = {2,
    3};
17.      p1 = foo();
18.      printf("%d\n", p1.x);
19.  }
20.  struct point foo()
21.  {
22.      struct point temp = {1,
    2};
23.      return temp;
24.  }
```

a) Compile time error
b) 1
c) 2
d) Undefined behaviour

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct point
3.  {
4.      int x;
5.      int y;
6.  };
7.  struct notpoint
8.  {
9.      int x;
10.     int y;
11.  };
12.  int main()
13.  {
14.      struct point p = {1};
15.      struct notpoint p1 = p;
16.      printf("%d\n", p1.x);
17.  }
```

a) Compile time error
b) 1
c) 0
d) Undefined

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct point
3.  {
4.      int x;
5.      int y;
6.  };
```

```
7.     struct notpoint
8.     {
9.         int x;
10.         int y;
11.     };
12.     void foo(struct point);
13.     int main()
14.     {
15.         struct notpoint p1 = {1,
    2};
16.         foo(p1);
17.     }
18.     void foo(struct point p)
19.     {
20.         printf("%d\n", p.x);
21.     }
```

a) Compile time error
b) 1
c) 0
d) Undefined

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct point
3.     {
4.         int x;
5.         int y;
6.     };
7.     void foo(struct point*);
8.     int main()
9.     {
10.         struct point p1 = {1,
    2};
11.         foo(&p1);
12.     }
13.     void foo(struct point *p)
14.     {
15.         printf("%d\n", *p.x++);
16.     }
```

a) Compile time error
b) Segmentation fault/code crash
c) 2
d) 1

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct point
3.     {
4.         int x;
5.         int y;
6.     };
7.     void foo(struct point*);
8.     int main()
9.     {
10.         struct point p1 = {1,
    2};
11.         foo(&p1);
12.     }
13.     void foo(struct point *p)
14.     {
15.         printf("%d\n", *p->x++);
16.     }
```

a) Compile time error
b) 1
c) Segmentation fault/code crash
d) 2

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct student fun(void)
3.     {
4.         struct student
5.         {
6.             char *name;
7.         };
8.         struct student s;
9.         s.name = "alan";
10.         return s;
11.     }
12.     void main()
13.     {
14.         struct student m =
    fun();
15.         printf("%s", m.name);
```

```
16.     }
```

a) Compile time error
b) alan
c) Nothing
d) Varies

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct student
3.     {
4.         char *name;
5.     };
6.     struct student fun(void)
7.     {
8.         struct student s;
9.         s.name = "alan";
10.         return s;
11.     }
12.     void main()
13.     {
14.         struct student m =
    fun();
15.         printf("%s", m.name);
16.     }
```

a) Nothing
b) alan
c) Run time error
d) Varies

ANSWER:- b
Explanation: None.

1. The correct syntax to access the member of the ith structure in the array of structures is?

```
Assuming: struct temp
    {
        int b;
    }s[50];
```

a) s.b.[i];
b) s.[i].b;
c) s.b[i];
d) s[i].b;

ANSWER:- d
Explanation: None.

2. Comment on the output of the following C code.

```
1.     #include <stdio.h>
2.     struct temp
3.     {
4.         int a;
5.         int b;
6.         int c;
7.     };
8.     main()
9.     {
10.         struct temp p[] = {{1,
    2, 3}, {4, 5, 6}, {7, 8, 9}};
11.     }
```

a) No Compile time error, generates an array of structure of size 3
b) No Compile time error, generates an array of structure of size 9
c) Compile time error, illegal declaration of a multidimensional array
d) Compile time error, illegal assignment to members of structure

ANSWER:- a
Explanation: None.

3. Which of the following uses structure?
a) Array of structures
b) Linked Lists
c) Binary Tree
d) All of the mentioned

ANSWER:- d
Explanation: None.

4. What is the correct syntax to declare a function foo() which receives an array of structure in function?
a) void foo(struct *var);
b) void foo(struct *var[]);
c) void foo(struct var);
d) none of the mentioned

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code? (Assuming size of int be 4)

```
#include <stdio.h>
struct temp
{
    int a;
    int b;
    int c;
} p[] = {0};
main()
{
     printf("%d", sizeof(p));
 }
```

a) 4
b) 12
c) 16
d) Can't be estimated due to ambiguous initialization of array

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *name;
};
struct student s[2];
void main()
{
    s[0].name = "alan";
     s[1] = s[0];
     printf("%s%s",
  s[0].name, s[1].name);
     s[1].name = "turing";
     printf("%s%s",
  s[0].name, s[1].name);
 }
```

a) alan alan alan turing
b) alan alan turing turing
c) alan turing alan turing
d) run time error

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *name;
};
struct student s[2], r[2];
void main()
{
    s[0].name = "alan";
     s[1] = s[0];
     r = s;
     printf("%s%s",
  r[0].name, r[1].name);
 }
```

a) alan alan
b) Compile time error
c) Varies
d) Nothing

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *name;
};
void main()
{
    struct student s[2],
  r[2];
    s[1] = s[0] = "alan";
     printf("%s%s",
  s[0].name, s[1].name);
 }
```

a) alan alan
b) Nothing
c) Compile time error
d) Varies

ANSWER:- c
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
struct student
```

```
3.     {
4.     };
5.     void main()
6.     {
7.         struct student s[2];
8.         printf("%d", sizeof(s));
9.     }
```

a) 2
b) 4
c) 8
d) 0

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct point
3.     {
4.         int x;
5.         int y;
6.     };
7.     void foo(struct point*);
8.     int main()
9.     {
10.         struct point p1[]  =
    {1, 2, 3, 4};
11.         foo(p1);
12.     }
13.     void foo(struct point p[])
14.     {
15.         printf("%d\n", p[1].x);
16.     }
```

a) Compile time error
b) 3
c) 2
d) 1

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct point
3.     {
4.         int x;
5.         int y;
6.     };
7.     void foo(struct point*);
8.     int main()
9.     {
10.         struct point p1[] = {1,
    2, 3, 4};
11.         foo(p1);
12.     }
13.     void foo(struct point p[])
14.     {
15.         printf("%d\n", p->x);
16.     }
```

a) 1
b) 2
c) 3
d) Compile time error

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     struct point
3.     {
4.         int x;
5.         int y;
6.     };
7.     void foo(struct point*);
8.     int main()
9.     {
10.         struct point p1[] = {1,
    2, 3, 4};
11.         foo(p1);
12.     }
13.     void foo(struct point p[])
14.     {
15.         printf("%d %d\n", p->x,
    ++p->x);
16.     }
```

a) 1 2
b) 2 2
c) Compile time error
d) Undefined behaviour

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
struct point
{
    int x;
    int y;
} p[] = {1, 2, 3, 4, 5};
void foo(struct point*);
int main()
{
     foo(p);
 }
 void foo(struct point p[])
 {
     printf("%d %d\n", p->x, p[2].y);
 }
```

a) 1 0
b) Compile time error
c) 1 somegarbagevalue
d) Undefined behaviour

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};
void foo(struct point*);
int main()
{
     struct point p1[] = {1, 2, 3, 4, 5};
     foo(p1);
 }
 void foo(struct point p[])
 {
     printf("%d %d\n", p->x, p[3].y);
 }
```

a) Compile time error
b) 1 0
c) 1 somegarbagevalue
d) None of the mentioned

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};
void foo(struct point*);
int main()
{
     struct point p1[] = {1, 2, 3, 4, 5};
     foo(p1);
 }
 void foo(struct point p[])
 {
     printf("%d %d\n", p->x, (p + 2).y);
 }
```

a) Compile time error
b) 1 0
c) 1 somegarbagevalue
d) Undefined behaviour

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};
void foo(struct point*);
int main()
{
     struct point p1[] = {1, 2, 3, 4, 5};
```

```
        foo(p1);
    }
    void foo(struct point p[])
    {
        printf("%d %d\n", p->x,
   (p + 2)->y);
    }
```

a) Compile time error
b) 1 0
c) 1 somegarbagevalue
d) undefined behaviour

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student s[2];
    printf("%d", sizeof(s));
 }
```

a) 2
b) 4
c) 16
d) 8

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x;
    char y;
};
int main()
{
    struct p p1[] = {1, 92,
   3, 94, 5, 96};
     struct p *ptr1 = p1;
    int x = (sizeof(p1) /
   3);
    if (x == sizeof(int) +
   sizeof(char))
        printf("%d\n",
   ptr1->x);
    else
        printf("falsen");
}
```

a) Compile time error
b) 1
c) Undefined behaviour
d) false

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x;
    char y;
};
int main()
{
    struct p p1[] = {1, 92,
   3, 94, 5, 96};
     struct p *ptr1 = p1;
    int x = (sizeof(p1) /
   sizeof(ptr1));
    if (x == 1)
        printf("%d\n",
   ptr1->x);
    else
        printf("false\n");
}
```

a) Compile time error
b) 1
c) false
d) Undefined behaviour

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x;
    char y;
};
typedef struct p* q*;
int main()
{
     struct p p1[] = {1, 92, 3, 94, 5, 96};
     q ptr1 = p1;
     printf("%d\n", ptr1->x);
 }
```

a) Compile time error
b) 1
c) Undefined behaviour
d) Segmentation fault

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x;
    char y;
};
void foo(struct p* );
int main()
{
     typedef struct p* q;
     struct p p1[] = {1, 92, 3, 94, 5, 96};
     foo(p1);
 }
 void foo(struct p* p1)
 {
     q ptr1 = p1;
     printf("%d\n", ptr1->x);
 }
```

a) Compile time error
b) 1
c) Segmentation fault
d) Undefined behaviour

ANSWER:- a
Explanation: None.

5. Which of the following is an incorrect syntax for pointer to structure?

(Assuming struct temp{int b;}*my_struct;)

a) *my_struct.b = 10;
b) (*my_struct).b = 10;
c) my_struct->b = 10;
d) Both *my_struct.b = 10; and (*my_struct).b = 10;

ANSWER:- a
Explanation: None.

6. Which of the following is an incorrect syntax to pass by reference a member of a structure in a function?

(Assume: struct temp{int a;}s;)

a) func(&s.a);
b) func(&(s).a);
c) func(&(s.a));
d) none of the mentioned

ANSWER:- d
Explanation: None.

7. Which of the following structure declaration doesn't require pass-by-reference?
a)

```
struct{int a;}s;
main(){}
```

b)

```
struct temp{int a;};
main(){
    struct temp s;
}
```

c)

```
struct temp{int a;};
main(){}
struct temp s;
```

d) none of the mentioned

ANSWER:- d
Explanation: None.

8. Which option is not possible for the following function call?

```
func(&s.a); //where s is a variable of type struct and a is the member of the struct.
```

a) Compiler can access entire structure from the function
b) Individual member's address can be displayed in structure
c) Individual member can be passed by reference in a function
d) None of the mentioned

ANSWER:- a
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
struct temp
{
    int a;
} s;
void change(struct temp);
main()
{
    s.a = 10;
     change(s);
     printf("%d\n", s.a);
 }
 void change(struct temp s)
 {
     s.a = 1;
 }
```

a) Output will be 1
b) Output will be 10
c) Output varies with machine
d) Compile time error

ANSWER:- b
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
struct p
 {
    int x;
    int y;
};
int main()
{
    struct p p1[] = {1, 92, 3, 94, 5, 96};
     struct p *ptr1 = p1;
     int x = (sizeof(p1) / 5);
     if (x == 3)
         printf("%d %d\n", ptr1->x, (ptr1 + x - 1)->x);
     else
         printf("false\n");
 }
```

a) Compile time error
b) 1 5
c) Undefined behaviour
d) false

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student m;
    struct student *s = &m;
     s->c = "hello";
     printf("%s", s->c);
 }
```

a) hello
b) Run time error
c) Nothing
d) Depends on compiler

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student *s;
    s->c = "hello";
     printf("%s", s->c);
 }
```

a) hello
b) Segmentation fault
c) Run time error
d) Nothing

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student m;
    struct student *s = &m;
     s->c = "hello";
     printf("%s", m.c);
 }
```

a) Run time error
b) Nothing
c) hello
d) Varies

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student m;
    struct student *s = &m;
     (*s).c = "hello";
     printf("%s", m.c);
 }
```

a) Run time error
b) Nothing
c) Varies
d) hello

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
struct student
{
    char *c;
};
void main()
{
    struct student n;
    struct student *s = &n;
     (*s).c = "hello";
     printf("%p\n%p\n", s,
  &n);
 }
```

a) Different address
b) Run time error
c) Nothing
d) Same address

ANSWER:- d
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x[2];
};
struct q
{
    int *x;
```

```
    };
     int main()
    {
        struct p p1 = {1, 2};
        struct q *ptr1;
        ptr1->x = (struct
q*)&p1.x;
        printf("%d\n", ptr1-
>x[1]);
    }
```

a) Compile time error
b) Segmentation fault/code crash
c) 2
d) 1

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
    #include <stdio.h>
    struct p
    {
        int x[2];
    };
    struct q
    {
        int *x;
    };
     int main()
    {
        struct p p1 = {1, 2};
        struct q *ptr1 =
(struct q*)&p1;
        ptr1->x = (struct
q*)&p1.x;
        printf("%d\n", ptr1-
>x[0]);
    }
```

a) Compile time error
b) Undefined behaviour
c) Segmentation fault/code crash
d) 1

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
    #include <stdio.h>
    struct p
    {
        int x;
        int y;
    };
    int main()
    {
        struct p p1[] = {1, 2, 3,
4, 5, 6};
         struct p *ptr1 = p1;
         printf("%d %d\n", ptr1-
>x, (ptr1 + 2)->x);
    }
```

a) 1 5
b) 1 3
c) Compile time error
d) 1 4

ANSWER:- a
Explanation: None.

9. What will be the output of the following C code?

```
    #include <stdio.h>
    struct p
    {
        int x;
        char y;
    };
    int main()
    {
        struct p p1[] = {1, 92,
3, 94, 5, 96};
         struct p *ptr1 = p1;
         int x = (sizeof(p1) /
sizeof(struct p));
         printf("%d %d\n", ptr1-
>x, (ptr1 + x - 1)->x);
    }
```

a) Compile time error
b) Undefined behaviour
c) 1 3
d) 1 5

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      char *c;
5.      struct student *point;
6.  };
7.  void main()
8.  {
9.      struct student s;
10.     struct student m;
11.     s.c = m.c = "hi";
12.     m.point = &s;
13.     (m.point)->c = "hey";
14.     printf("%s\t%s\t", s.c,
    m.c);
15.  }
```

a) hey hi
b) hi hey
c) Run time error
d) hey hey

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      char *c;
5.      struct student *point;
6.  };
7.  void main()
8.  {
9.      struct student s;
10.     struct student m;
11.     m.point = s;
12.     (m.point)->c = "hey";
13.     printf("%s", s.c);
14.  }
```

a) Nothing
b) Compile time error
c) hey
d) Varies

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      char *c;
5.      struct student point;
6.  };
7.  void main()
8.  {
9.      struct student s;
10.     s.c = "hello";
11.     printf("%s", s.c);
12.  }
```

a) hello
b) Nothing
c) Varies
d) Compile time error

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      char *c;
5.      struct student *point;
6.  };
7.  void main()
8.  {
9.      struct student s;
10.     printf("%d", sizeof(s));
11.  }
```

a) 5
b) 9
c) 8
d) 16

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct student
3.  {
4.      char *c;
5.      struct student *point;
```

```
};
void main()
{
    struct student s;
     struct student *m = &s;
     printf("%d",
sizeof(student));
 }
```

a) Compile time error
b) 8
c) 5
d) 16

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    int x;
    char y;
    struct p *ptr;
};
int main()
{
     struct p p = {1, 2, &p};
     printf("%d\n", p.ptr-
>x);
     return 0;
 }
```

a) Compile time error
b) Undefined behaviour
c) 1
d) 2

ANSWER:- c
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct p *q;
struct p
{
    int x;
    char y;
    q ptr;
};
int main()
 {
     struct p p = {1, 2, &p};
     printf("%d\n", p.ptr-
>x);
     return 0;
 }
```

a) Compile time error
b) 1
c) Undefined behaviour
d) Address of p

ANSWER:- b
Explanation: None.

8. Presence of loop in a linked list can be tested by _____
a) Traveling the list, if NULL is encountered no loop exists
b) Comparing the address of nodes by address of every other node
c) Comparing the the value stored in a node by a value in every other node
d) None of the mentioned

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct p *q;
int main()
{
    struct p
    {
        int x;
        char y;
        q ptr;
     };
     struct p p = {1, 2, &p};
     printf("%d\n", p.ptr-
>x);
     return 0;
 }
```

a) Compile time error
b) 1
c) Depends on the compiler

d) None of the mentioned

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    typedef struct p *q;
    struct p
    {
        int x;
        char y;
        q ptr;
     };
     struct p p = {1, 2, &p};
     printf("%d\n", p.ptr->x);
     return 0;
 }
```

a) Compile time error
b) 1
c) Depends on the compiler
d) Depends on the standard

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct p *q;
struct p
{
    int x;
    char y;
    q ptr;
};
int main()
 {
     struct p p = {1, 2, &p};
     printf("%d\n", p.ptr->ptr->x);
     return 0;
 }
```

a) Compile time error
b) Segmenation fault
c) Undefined behaviour
d) 1

ANSWER:- d
Explanation: None.

4. The number of distinct nodes the following struct declaration can point to is _____

```
struct node
{
    struct node *left;
    struct node *centre;
    struct node *right;
};
```

a) 1
b) 2
c) 3
d) All of the mentioned

ANSWER:- d
Explanation: None.

5. Which of the following is not possible regarding the structure variable?
a) A structure variable pointing to itself
b) A structure variable pointing to another structure variable of same type
c) 2 different type of structure variable pointing at each other
d) None of the mentioned

ANSWER:- d
Explanation: None.

6. Which of the following technique is faster for travelling in binary trees?
a) Iteration
b) Recursion
c) Both Iteration and Recursion
d) Depends from compiler to compiler

ANSWER:- b
Explanation: None.

7. Which of the following will stop the loop at the last node of a linked list in the following C code snippet?

```
struct node
{
    struct node *next;
```

```
4.      };
```

a)

```
while (p != NULL)
{
    p = p->next;
}
```

b)

```
while (p->next != NULL)
{
    p = p->next;
}
```

c)

```
while (1)
{
    p = p->next;
    if (p == NULL)
        break;
}
```

d) All of the mentioned

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      struct student
3.      {
4.          char a[5];
5.      };
6.      void main()
7.      {
8.          struct student s[] =
    {"hi", "hey"};
9.          printf("%c", s[0].a[1]);
10.     }
```

a) h
b) i
c) e
d) y

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          char *a[3] = {"hello",
    "this"};
5.          printf("%s", a[1]);
6.      }
```

a) hello
b) Varies
c) this
d) Compile time error

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          int lookup[100] = {0, 1,
    2, 3, 4, 5, 6, 7, 8, 9};
5.          printf("%d", lookup[3]);
6.      }
```

a) 2
b) 4
c) Compile time error
d) 3

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      void main()
3.      {
4.          char *a[3][3] = {{"hey",
    "hi", "hello"}, {"his", "her",
    "hell"}
5.          , {"hellos", "hi's",
    "hmm"}};
6.          printf("%s", a[1][1]);
7.      }
```

a) her
b) hi
c) Compile time error

d) hi's

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    char *name;
    struct p *next;
};
struct p *ptrary[10];
int main()
{
     struct p p;
     p->name = "xyz";
     p->next = NULL;
     ptrary[0] = &p;
     printf("%s\n", p->name);
     return 0;
 }
```

a) Compile time error
b) Segmentation fault/code crash
c) xyz
d) Undefined behaviour

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    char *name;
    struct p *next;
};
struct p *ptrary[10];
int main()
{
     struct p p;
     p.name = "xyz";
     p.next = NULL;
     ptrary[0] = &p;
     printf("%s\n",
ptrary[0]->name);
     return 0;
 }
```

a) Compile time error
b) Segmentation fault
c) Undefined behaviour
d) xyz

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    char *name;
    struct p *next;
};
struct p *ptrary[10];
int main()
{
     struct p p, q;
     p.name = "xyz";
     p.next = NULL;
     ptrary[0] = &p;
     strcpy(q.name, p.name);
     ptrary[1] = &q;
     printf("%s\n",
ptrary[1]->name);
     return 0;
 }
```

a) Compile time error
b) Segmentation fault/code crash
c) Depends on the compiler
d) xyz

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    struct p
    {
        char *name;
        struct p *next;
```

```
8.          };
9.          struct p p, q;
10.          p.name = "xyz";
11.          p.next = NULL;
12.          ptrary[0] = &p;
13.          strcpy(q.name, p.name);
14.          ptrary[1] = &q;
15.          printf("%s\n",
    ptrary[1]->name);
16.          return 0;
17.      }
```

a) Compile time error
b) Depends on the compiler
c) Undefined behaviour
d) xyz

ANSWER:- c
Explanation: None.

1. Which function is responsible for searching in the table? (For #define IN 1, the name IN and replacement text 1 are stored in a "table")
a) findout(s);
b) lookup(s);
c) find(s);
d) lookfor(s);

ANSWER:- b
Explanation: None.

2. Which algorithm is used for searching in the table?
a) List search
b) Informed search
c) Hash search
d) Adversarial search

ANSWER:- c
Explanation: None.

3. Which function is responsible for recording the name "s" and the replacement text "t" in a table?
a) install(s, t);
b) fix(s, t);
c) setup(s, t);
d) settle(s, t);

ANSWER:- a
Explanation: None.

4. Which of the following statement is true?
a) Install function uses lookup
b) lookup function uses install
c) Install and lookup function work independently
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. What happens when install(s, t) finds that the name being installed is already present in the table?
a) It doesn't modify the name in the table
b) It modifies the name with new definition
c) It modifies off the new definition has higher priority
d) It creates a new table and add the new definition in it

ANSWER:- b
Explanation: None.

6. In what situation, install function returns NULL?
a) When there is no memory for adding new name
b) When the name to be defined is already present in the table
c) Whenever a new name is added to the table
d) All of the mentioned

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct student
3.    {
4.        char a[];
5.    };
6.    void main()
7.    {
8.        struct student s;
9.        printf("%d",
    sizeof(struct student));
10.    }
```

a) Compile time error
b) 8
c) 1
d) Varies

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    struct p
    {
        char *name;
        struct p *next;
    };
    struct p *ptrary[10];
     struct p p, q;
     p.name = "xyz";
     p.next = NULL;
     ptrary[0] = &p;
     q.name =
(char*)malloc(sizeof(char)*3);
     strcpy(q.name, p.name);
     q.next = &q;
     ptrary[1] = &q;
     printf("%s\n",
ptrary[1]->next->next->name);
 }
```

a) Compile time error
b) Depends on the compiler
c) Undefined behaviour
d) xyz

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct student
{
    char *a;
}stu;
void main()
{
    struct stu s;
    s.a = "hi";
     printf("%s", s.a);
 }
```

a) Compile time error
b) Varies
c) hi
d) h

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct student
{
    char *a;
}stu;
void main()
{
    struct student s;
    s.a = "hey";
     printf("%s", s.a);
 }
```

a) Compile time error
b) Varies
c) he
d) hey

ANSWER:- d
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
typedef int integer;
int main()
{
    int i = 10, *ptr;
    float f = 20;
    integer j = i;
    ptr = &j;
    printf("%d\n", *ptr);
     return 0;
 }
```

a) Compile time error
b) Undefined behaviour
c) Depends on the standard
d) 10

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int (*(x()))[2];
```

```
3.    typedef int (*(*ptr)())[2]
   ptrfoo;
4.    int main()
5.    {
6.        ptrfoo ptr1;
7.        ptr1 = x;
8.        ptr1();
9.        return 0;
10.    }
11.    int (*(x()))[2]
12.    {
13.        int (*ary)[2] =
   malloc(sizeof*ary);
14.        return &ary;
15.    }
```

a) Compile time error
b) Nothing
c) Undefined behaviour
d) Depends on the standard

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int *(*(x()))[2];
3.    typedef int
   **(*ptrfoo)())[2];
4.    int main()
5.    {
6.        ptrfoo ptr1;
7.        ptr1 = x;
8.        ptr1();
9.        return 0;
10.    }
11.    int *(*(x()))[2]
12.    {
13.        int (*ary)[2] =
   malloc(sizeof * ary);
14.        return &ary;
15.    }
```

a) Compile time error
b) Nothing
c) Undefined behaviour
d) Depends on the standard

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    typedef struct p
3.    {
4.        int x, y;
5.    };
6.    int main()
7.    {
8.        p k1 = {1, 2};
9.        printf("%d\n", k1.x);
10.    }
```

a) Compile time error
b) 1
c) 0
d) Depends on the standard

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    typedef struct p
3.    {
4.        int x, y;
5.    }k = {1, 2};
6.    int main()
7.    {
8.        p k1 = k;
9.        printf("%d\n", k1.x);
10.    }
```

a) Compile time error
b) 1
c) 0
d) Depends on the standard

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    typedef struct p
3.    {
4.        int x, y;
```

```
5.     }k;
6.     int main()
7.     {
8.         struct p p = {1, 2};
9.         k k1 = p;
10.         printf("%d\n", k1.x);
11.     }
```

a) Compile time error
b) 1
c) 0
d) Depends on the standard

ANSWER:- b
Explanation: None.

1. Which is the correct syntax to use typedef for struct?
a)

```
typedef struct temp
{
    int a;
}TEMP;
```

b)

```
typedef struct
{
    int a;
 }TEMP;
```

c)

```
struct temp
{
    int a;
};
typedef struct temp TEMP;
```

d) All of the mentioned

ANSWER:- d
Explanation: None.

2. Which option should be selected to work the following C expression?

```
string p = "HELLO";
```

a) typedef char [] string;
b) typedef char *string;
c) typedef char [] string; and typedef char *string;
d) Such expression cannot be generated in C

ANSWER:- b
Explanation: None.

3. Which of the given option is the correct method for initialization?

```
typedef char *string;
```

a) *string *p = “Hello”;
b) string p = “Hello”;
c) *string p = ‘A’;
d) Not more than one space should be given when using typedef

ANSWER:- b
Explanation: None.

4. Which of the following is false about typedef?
a) typedef follow scope rules
b) typedef defined substitutes can be redefined again. (Eg: typedef char a; typedef int a;)
c) You cannot typedef a typedef with other term
d) All of the mentioned

ANSWER:- b
Explanation: None.

5. Which of the following may create problem in the typedef program?
a) ;
b) printf/scanf
c) Arithmetic operators
d) All of the mentioned

ANSWER:- d
Explanation: None.

6. typedef int (*PFI)(char *, char *)creates _____
a) type PFI, for pointer to function (of two char * arguments) returning int
b) error
c) type PFI, function (of two char * arguments) returning int
d) type PFI, for pointer

ANSWER:- a
Explanation: None.

7. What is typedef declaration?
a) Does not create a new type
b) It merely adds a new name for some existing type
c) Does not create a new type, It merely adds a new name for some existing type

d) None of the mentioned

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
typedef struct student
{
    char *a;
}stu;
void main()
{
    stu s;
    s.a = "hi";
     printf("%s", s.a);
 }s
```

a) Compile time error
b) Varies
c) hi
d) h

ANSWER:- a
Explanation: None.

1. The size of a union is determined by the size of the _____
a) First member in the union
b) Last member in the union
c) Biggest member in the union
d) Sum of the sizes of all members

ANSWER:- c
Explanation: None.

2. Which member of the union will be active after REF LINE in the following C code?

```
#include <stdio.h>
union temp
{
    int a;
    float b;
    char c;
};
union temp s = {1,2.5,'A'};
//REF LINE
```

a) a
b) b
c) c
d) Such declaration are illegal

ANSWER:- a
Explanation: None.

3. What would be the size of the following union declaration? (Assuming size of double = 8, size of int = 4, size of char = 1)

```
#include <stdio.h>
union uTemp
{
    double a;
    int b[10];
    char c;
}u;
```

a) 4
b) 8
c) 40
d) 80

ANSWER:- c
Explanation: None.

4. What type of data is holded by variable u int in the following C code?

```
#include <stdio.h>
union u_tag
{
    int ival;
    float fval;
    char *sval;
} u;
```

a) Will be large enough to hold the largest of the three types;
b) Will be large enough to hold the smallest of the three types;
c) Will be large enough to hold the all of the three types;
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. Members of a union are accessed as_____
a) union-name.member
b) union-pointer->member
c) both union-name.member & union-pointer->member

d) none of the mentioned

ANSWER:- c
Explanation: None.

6. In the following C code, we can access the 1st character of the string sval by using _____

```
#include <stdio.h>
struct
{
    char *name;
    union
    {
        char *sval;
    } u;
} symtab[10];
```

a) *symtab[i].u.sval
b) symtab[i].u.sval[0].
c) You cannot have union inside structure
d) Both *symtab[i].u.sval & symtab[i].u.sval[0].

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code (Assuming size of int and float is 4)?

```
#include <stdio.h>
union
{
    int ival;
    float fval;
} u;
void main()
{
    printf("%d", sizeof(u));
 }
```

a) 16
b) 8
c) 4
d) 32

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
union stu
{
    int ival;
    float fval;
};
void main()
{
    union stu r;
     r.ival = 5;
     printf("%d", r.ival);
 }
```

a) 9
b) Compile time error
c) 16
d) 5

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
union
{
    int x;
    char y;
}p;
int main()
{
    p.x = 10;
     printf("%d\n",
sizeof(p));
 }
```

a) Compile time error
b) sizeof(int) + sizeof(char)
c) Depends on the compiler
d) sizeof(int)

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
union
{
    int x;
    char y;
}p;
```

```
7.     int main()
8.     {
9.         p.y = 60;
10.         printf("%d\n",
    sizeof(p));
11.     }
```

a) Compile time error
b) sizeof(int) + sizeof(char)
c) Depends on the compiler
d) sizeof(char)

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     union p
3.     {
4.         int x;
5.         char y;
6.     };
7.     int main()
8.     {
9.         union p p, b;
10.         p.y = 60;
11.         b.x = 12;
12.         printf("%d\n", p.y);
13.     }
```

a) Compile time error
b) Depends on the compiler
c) 60
d) Undefined behaviour

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     union p
3.     {
4.         int x;
5.         char y;
6.     }k = {1, 97};
7.     int main()
8.     {
9.         printf("%d\n", k.y);
10.     }
```

a) Compile time error
b) 97
c) a
d) 1

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     union p
3.     {
4.         int x;
5.         char y;
6.     }k = {.y = 97};
7.     int main()
8.     {
9.         printf("%d\n", k.y);
10.     }
```

a) Compile time error
b) 97
c) a
d) Depends on the standard

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.     #include <stdio.h>
2.     union p
3.     {
4.         int x;
5.         float y;
6.     };
7.     int main()
8.     {
9.         union p p, b;
10.         p.x = 10;
11.         printf("%f\n", p.y);
12.     }
```

a) Compile time error
b) Implementation dependent
c) 10.000000
d) 0.000000

ANSWER:- b
Explanation: None.

7. Which of the following share a similarity in syntax?

1. Union, 2. Structure, 3. Arrays and 4. Pointers

a) 3 and 4
b) 1 and 2
c) 1 and 3
d) 1, 3 and 4

ANSWER:- b
Explanation: None.

8. What will be the output of the following C code? (Assuming size of char = 1, int = 4, double = 8)

```
1.  #include <stdio.h>
2.  union utemp
3.  {
4.      int a;
5.      double b;
6.      char c;
7.  }u;
8.  int main()
9.  {
10.      u.c = 'A';
11.      u.a = 1;
12.      printf("%d", sizeof(u));
13.  }
```

a) 1
b) 4
c) 8
d) 13

ANSWER:- c
Explanation: None.

1. What is the correct syntax to initialize bit-fields in an structure?
a)

```
struct temp
{
    unsigned int a : 1;
}s;
```

b)

```
struct temp
{
    unsigned int a = 1;
}s;
```

c)

```
struct temp
{
    unsigned float a : 1;
}s;
```

d) None of the mentioned

ANSWER:- a
Explanation: None.

2. Which of the following data types are accepted while declaring bit-fields?
a) char
b) float
c) double
d) none of the mentioned

ANSWER:- a
Explanation: None.

3. Which of the following reduces the size of a structure?
a) union
b) bit-fields
c) malloc
d) none of the mentioned

ANSWER:- b
Explanation: None.

4. For what minimum value of x in a 32-bit Linux OS would make the size of s equal to 8 bytes?

```
1.  struct temp
2.  {
3.      int a : 13;
4.      int b : 8;
5.      int c : x;
6.  }s;
```

a) 4
b) 8
c) 12
d) 32

ANSWER:- c
Explanation: None.

5. Calculate the % of memory saved when bit-fields are used for the following C structure as compared to with-out use of bit-fields for the same structure? (Assuming size of int = 4)

```
1.    struct temp
2.    {
3.        int a : 1;
4.        int b : 2;
5.        int c : 4;
6.        int d : 4;
7.    }s;
```

a) 25%
b) 33.3%
c) 50%
d) 75%

ANSWER:- d
Explanation: None.

6. In the following declaration of bit-fields, the constant-expression specifies _____

```
   struct-declarator:
   declarator
   type-specifier declarator opt :
constant-expression
```

a) The width of the field in bits
b) Nothing
c) The width of the field in bytes
d) Error

ANSWER:- a
Explanation: None.

7. In the following declaration of bit-fields, the constant-expression must be _____

```
   struct-declarator:
   declarator
   type-specifier declarator opt :
constant-expression
```

a) Any type
b) Nothing
c) Integer value
d) Nonnegative integer value

ANSWER:- d
Explanation: None.

8. Which of the following is not allowed?
a) Arrays of bit fields
b) Pointers to bit fields
c) Functions returning bit fields
d) None of the mentioned

ANSWER:- d
Explanation: None.

9. Bit fields can only be declared as part of a structure.
a) false
b) true
c) Nothing
d) Varies

ANSWER:- b
Explanation: None.

10. What is the order for the following C declarations?

```
   short a : 17;
   int long y : 33;
```

a) Legal, legal
b) Legal, illegal
c) Illegal, illegal
d) Illegal, legal

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct p
3.    {
4.        char x : 2;
5.        int y : 2;
6.    };
7.    int main()
8.    {
9.        struct p p;
10.        p.x = 2;
11.        p.y = 1;
12.        p.x = p.x & p.y;
13.        printf("%d\n", p.x);
14.    }
```

a) 0
b) Compile time error
c) Undefined behaviour

d) Depends on the standard

ANSWER:- a
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
union u
{
    struct p
    {
        unsigned char x : 2;
        unsigned int y : 2;
    };
    int x;
 };
 int main()
 {
     union u u;
     u.p.x = 2;
     printf("%d\n", u.p.x);
 }
```

a) Compile time error
b) Undefined behaviour
c) Depends on the standard
d) 2

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
union u
{
    struct
    {
        unsigned char x : 2;
        unsigned int y : 2;
    }p;
    int x;
 };
 int main()
 {
     union u u;
     u.p.x = 2;
     printf("%d\n", u.p.x);
 }
```

a) Compile time error
b) 2
c) Undefined behaviour
d) Depends on the standard

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
union u
{
    struct
    {
        unsigned char x : 2;
        unsigned int y : 2;
    }p;
    int x;
 };
 int main()
 {
     union u u.p.x = 2;
     printf("%d\n", u.p.x);
 }
```

a) Compile time error
b) 2
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
union u
{
    struct
    {
        unsigned char x : 2;
        unsigned int y : 2;
    }p;
    int x;
 };
 int main()
 {
```

```
13.         union u u = {2};
14.         printf("%d\n", u.p.x);
15.     }
```

a) Compile time error
b) 2
c) Depends on the standard
d) None of the mentioned

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  union u
3.  {
4.      struct
5.      {
6.          unsigned char x : 2;
7.          unsigned int y : 2;
8.      }p;
9.      int x;
10.  };
11.  int main()
12.  {
13.      union u u.p = {2};
14.      printf("%d\n", u.p.x);
15.  }
```

a) Compile time error
b) 2
c) Undefined behaviour
d) None of the mentioned

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct p
3.  {
4.      unsigned int x : 2;
5.      unsigned int y : 2;
6.  };
7.  int main()
8.  {
9.      struct p p;
10.      p.x = 3;
11.      p.y = 1;
12.      printf("%d\n",
    sizeof(p));
13.  }
```

a) Compile time error
b) Depends on the compiler
c) 2
d) 4

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct p
3.  {
4.      unsigned int x : 2;
5.      unsigned int y : 2;
6.  };
7.  int main()
8.  {
9.      struct p p;
10.      p.x = 3;
11.      p.y = 4;
12.      printf("%d\n", p.y);
13.  }
```

a) 0
b) 4
c) Depends on the compiler
d) 2

ANSWER:- a
Explanation: None.

9. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  struct p
3.  {
4.      unsigned int x : 7;
5.      unsigned int y : 2;
6.  };
7.  int main()
8.  {
9.      struct p p;
10.      p.x = 110;
11.      p.y = 2;
```

```
12.         printf("%d\n", p.x);
13.     }
```

a) Compile time error
b) 110
c) Depends on the standard
d) None of the mentioned

ANSWER:- b
Explanation: None.

10. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    struct p
3.    {
4.        unsigned int x : 1;
5.        unsigned int y : 1;
6.    };
7.    int main()
8.    {
9.        struct p p;
10.        p.x = 1;
11.        p.y = 2;
12.        printf("%d\n", p.y);
13.    }
```

a) 1
b) 2
c) 0
d) Depends on the compiler

ANSWER:- c
Explanation: None.

1. Which among the following is the odd one out?
a) printf
b) fprintf
c) putchar
d) scanf

ANSWER:- d
Explanation: None.

2. For a typical program, the input is taken using _____
a) scanf
b) Files
c) Command-line
d) All of the mentioned

ANSWER:- d
Explanation: None.

3. What does the following command line signify?

```
prog1|prog2
```

a) It runs prog1 first, prog2 second
b) It runs prog2 first, prog1 second
c) It runs both the programs, pipes output of prog1 to input of prog2
d) It runs both the programs, pipes output of prog2 to input of prog1

ANSWER:- c
Explanation: None.

4. What is the default return-type of getchar()?
a) char
b) int
c) char *
d) reading character doesn't require a return-type

ANSWER:- b
Explanation: None.

5. What is the value of EOF?
a) -1
b) 0
c) 1
d) 10

ANSWER:- a
Explanation: None.

6. What is the use of getchar()?
a) The next input character each time it is called
b) EOF when it encounters end of file
c) The next input character each time it is called EOF when it encounters end of file
d) None of the mentioned

ANSWER:- c
Explanation: None.

7. Which of the following statement is true?
a) The symbolic constant EOF is defined in <stdio.h>
b) The value is -1
c) The symbolic constant EOF is defined in <stdio.h> & value is -1
d) Only value is -1

ANSWER:- c
Explanation: None.

8. What is the return value of putchar()?
a) The character written
b) EOF if an error occurs
c) Nothing

d) Both character written & EOF if an error occurs

ANSWER:- d
Explanation: None.

1. Which is true about function tolower?
a) The function tolower is defined in <ctype.h>
b) Converts an uppercase letter to lowercase
c) Returns other characters untouched
d) None of the mentioned

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char c = '';
    putchar(c);
}
```

a) Compile time error
b) Nothing
c) 0
d) Undefined behaviour

ANSWER:- b
Explanation: None.

3. putchar(c) function/macro always outputs character c to the _____
a) screen
b) standard output
c) depends on the compiler
d) depends on the standard

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code if following commands are used to run (considering myfile exists)?

```
gcc -otest test.c
./test < myfile

#include <stdio.h>
int main()
{
    char c = 'd';
    putchar(c);
}
```

a) Compile time error (after first command)
b) d in the myfile file
c) d on the screen
d) Undefined behaviour

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code if following commands are used to run (considering myfile exists)?

```
gcc -otest test.c
./test > myfile

 #include <stdio.h>
int main(int argc, char
**argv)
{
    char c = 'd';
    putchar(c);
    printf(" %d\n", argc);
 }
```

a) d 2 in myfile
b) d 1 in myfile
c) d in myfile and 1 in screen
d) d in myfile and 2 in screen

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code if following commands are used to run and if myfile does not exist?

```
gcc -o test test.c
./test > myfile

#include <stdio.h>
int main(int argc, char
**argv)
{
    char c = 'd';
    putchar(c);
    printf(" %d\n", argc);
 }
```

a) d 2 in myfile
b) d 1 in myfile
c) Depends on the system

d) Depends on the standard

ANSWER:- b
Explanation: None.

7. The statement prog < infile causes _____
a) prog to read characters from infile
b) prog to write characters to infile
c) infile to read characters from prog instead
d) nothing

ANSWER:- a
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10, j = 2;
5.        printf("%d\n",
    printf("%d %d ", i, j));
6.    }
```

a) Compile time error
b) 10 2 4
c) 10 2 2
d) 10 2 5

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10, j = 3;
5.        printf("%d %d %d", i, j);
6.    }
```

a) Compile time error
b) 10 3
c) 10 3 some garbage value
d) Undefined behaviour

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10, j = 3, k = 3;
5.        printf("%d %d ", i, j,
    k);
6.    }
```

a) Compile time error
b) 10 3 3
c) 10 3
d) 10 3 somegarbage value

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *s = "myworld";
5.        int i = 9;
6.        printf("%*s", i, s);
7.    }
```

a) myworld
b) myworld(note: spaces to the left of myworld)
c) myworld (note:followed by two spaces after myworld)
d) Undefined

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main(int argc, char**
    argv)
3.    {
4.        char *s = "myworld";
5.        int i = 3;
6.        printf("%10.*s", i, s);
7.    }
```

a) myw(note:7 spaces before myw)
b) myworld(note:2 spaces before myworld)
c) myworld (note:2 spaces after myworld)
d) myw(note:6 spaces after myw)

ANSWER:- a
Explanation: In the format represented by "%10.*s", the width of the string will be 10 spaces which is aligned to the right, by default. Since we have asterisk (*) after the precision dot (.), the

value of precision will be the value stored in the variable i. The value of i is 3, so this signifies max length of the string as 3 characters. So, the final formatted output will be a 10 character output with 3 characters “myw” printed with right alignment and the 1st 7 characters will be simply space characters.

6. What is the difference between %e and %g?
a) %e output formatting depends on the argument and %g always formats in the format [-]m.dddddd or [-]m.ddddd E[+|-]xx where no.of ds are optional
b) %e always formats in the format [-]m.dddddd or [-]m.ddddd E[+|-]xx where no.of ds are optional and output formatting depends on the argument
c) No differences
d) Depends on the standard

ANSWER:- b
Explanation: None.

7. Escape sequences are prefixed with _____
a) %
b) /
c) ”
d) None of the mentioned

ANSWER:- d
Explanation: None.

8. What is the purpose of sprintf?
a) It prints the data into stdout
b) It writes the formatted data into a string
c) It writes the formatted data into a file
d) None of the mentioned

ANSWER:- b
Explanation: None.

9. The syntax to print a % using printf statement can be done by _____
a) %
b) \%
c) ‘%’
d) %%

ANSWER:- d
Explanation: None.

1. What is the meaning of the following C statement?

printf(“%10s”, state);

a) 10 spaces before the string state is printed
b) Print empty spaces if the string state is less than 10 characters
c) Print the last 10 characters of the string
d) None of the mentioned

ANSWER:- b
Explanation: None.

2. What are the Properties of the first argument of a printf() functions?
a) It is defined by a user
b) It keeps the record of the types of arguments that will follow
c) There may no be first argument
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10, j = 2;
5.        printf("%d\n",
   printf("%d %d ", i, j));
6.    }
```

a) Compile time error
b) 10 2 4
c) 10 2 2
d) 10 2 5

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        int i = 10, j = 3;
5.        printf("%d %d %d", i, j);
6.    }
```

a) Compile time error
b) 10 3
c) 10 3 some garbage value
d) Undefined behaviour

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int i = 10, j = 3, k = 3;
    printf("%d %d ", i, j,
k);
}
```

a) Compile time error
b) 10 3 3
c) 10 3
d) 10 3 somegarbage value

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *s = "myworld";
    int i = 9;
    printf("%*s", i, s);
}
```

a) myworld
b) myworld(note: spaces to the left of myworld)
c) myworld (note:followed by two spaces after myworld)
d) Undefined

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main(int argc, char
**argv)
{
    char *s = "myworld";
    int i = 3;
    printf("%10.*s", i, s);
}
```

a) myw(note:7 spaces before myw)
b) myworld(note:2 spaces before myworld)
c) myworld (note:2 spaces after myworld)
d) myw(note:6 spaces after myworld)

ANSWER:- a
Explanation: In the format represented by "%10.*s", the width of the string will be 10 spaces which is aligned to the right, by default. Since we have asterisk (*) after the precision dot (.), the value of precision will be the value stored in the variable i. The value of i is 3, so this signifies max length of the string as 3 characters. So, the final formatted output will be a 10 character output with 3 characters "myw" printed with right alignment and the 1st 7 characters will be simply space characters.

8. What is the difference between %e and %g?
a) %e output formatting depends on the argument and %g always formats in the format [-]m.dddddd or [-]m.ddddd[+|-]xx where no.of ds are optional
b) %e always formats in the format [-]m.dddddd or [-]m.ddddd[+|-]xx where no.of ds are optional and output formatting depends on the argument
c) No differences
d) Depends on the standard

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
#include <stdarg.h>
void func(int, ...);
int main()
{
    func(2, 3, 5, 7, 11, 13);
    return 0;
}
void func(int n, ...)
 {
     int number, i = 0;
     va_list start;
     va_start(start, n);
     while (i != 3)
     {
         number =
va_arg(start, int);
         i++;
     }
     printf("%d", number);
```

```
20.     }
```

a) 3
b) 5
c) 7
d) 11

ANSWER:- c
Explanation: None.

2. Which of the following function with ellipsis are illegal?
a) void func(...);
b) void func(int, ...);
c) void func(int, int, ...);
d) none of the mentioned

ANSWER:- a
Explanation: None.

3. Which of the following data-types are promoted when used as a parameter for an ellipsis?
a) char
b) short
c) int
d) none of the mentioned

ANSWER:- a
Explanation: None.

4. Which header file includes a function for variable number of arguments?
a) stdlib.h
b) stdarg.h
c) ctype.h
d) both stdlib.h and stdarg.h

ANSWER:- b
Explanation: None.

5. Which of the following macro extracts an argument from the variable argument list (ie ellipsis) and advance the pointer to the next argument?
a) va_list
b) va_arg
c) va_end
d) va_start

ANSWER:- b
Explanation: None.

6. The type va_list in an argument list is used _____
a) To declare a variable that will refer to each argument in turn;
b) For cleanup
c) To create a list
d) There is no such type

ANSWER:- a
Explanation: None.

7. In a variable length argument function, the declaration "..." can _____
a) Appear anywhere in the function declaration
b) Only appear at the end of an argument list
c) Nothing
d) None of the mentioned

ANSWER:- b
Explanation: None.

8. Each call of va_arg _____
a) Returns one argument
b) Steps va_list variable to the next
c) Returns one argument & Steps va_list variable to the next
d) None of the mentioned

ANSWER:- c
Explanation: None.

1. The standard header _____ is used for variable list arguments (...) in C.
a) <stdio.h >
b) <stdlib.h>
c) <math.h>
d) <stdarg.h>

ANSWER:- d
Explanation: None.

2. What is the purpose of va_end?
a) Cleanup is necessary
b) Must be called before the program returns
c) Cleanup is necessary & Must be called before the program returns
d) None of the mentioned

ANSWER:- c
Explanation: None.

3. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  int f(char chr, ...);
3.  int main()
4.  {
5.      char c = 97;
6.      f(c);
```

```
7.          return 0;
8.      }
9.      int f(char c, ...)
10.      {
11.          printf("%c\n", c);
12.      }
```

a) Compile time error
b) Undefined behaviour
c) 97
d) a

ANSWER:- d
Explanation: None.

Take C Programming Tests Now!

4. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #include <stdarg.h>
3.      int f(...);
4.      int main()
5.      {
6.          char c = 97;
7.          f(c);
8.          return 0;
9.      }
10.      int f(...)
11.      {
12.          va_list li;
13.          char c = va_arg(li,
    char);
14.          printf("%c\n", c);
15.      }
```

a) Compile time error
b) Undefined behaviour
c) 97
d) a

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #include <stdarg.h>
3.      int f(char c, ...);
4.      int main()
5.      {
6.          char c = 97, d = 98;
7.          f(c, d);
8.          return 0;
9.      }
10.      int f(char c, ...)
11.      {
12.          va_list li;
13.          va_start(li, c);
14.          char d = va_arg(li,
    char);
15.          printf("%c\n", d);
16.          va_end(li);
17.      }
```

a) Compile time error
b) Undefined behaviour
c) a
d) b

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      #include <stdarg.h>
3.      int f(char c, ...);
4.      int main()
5.      {
6.          char c = 97, d = 98;
7.          f(c, d);
8.          return 0;
9.      }
10.      int f(char c, ...)
11.      {
12.          va_list li;
13.          va_start(li, c);
14.          char d = va_arg(li,
    int);
15.          printf("%c\n", d);
16.          va_end(li);
17.      }
```

a) Compile time error
b) Undefined behaviour
c) a
d) b

ANSWER:- d
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
#include <stdarg.h>
int f(int c, ...);
int main()
{
    int c = 97;
    float d = 98;
    f(c, d);
    return 0;
 }
 int f(int c, ...)
 {
     va_list li;
     va_start(li, c);
     float d = va_arg(li, float);
     printf("%f\n", d);
     va_end(li);
 }
```

a) Compile time error
b) Undefined behaviour
c) 97.000000
d) 98.000000

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", n);
    printf("%d\n", n);
    return 0;
}
```

a) Compilation error
b) Undefined behavior
c) Whatever user types
d) Depends on the standard

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char *n;
    scanf("%s", n);
    return 0;
}
```

a) Compilation error
b) Undefined behavior
c) Nothing
d) None of the mentioned

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char n[] = "hello\nworld!";
    char s[13];
    sscanf(n, "%s", s);
    printf("%s\n", s);
    return 0;
}
```

a) hellonworld!
b)

hello

world!

c) hello
d) hello world!

ANSWER:- c
Explanation: The array **n** contains a string which has a newline character in between the strings “hello” and “world”. A newline character is considered as a whitespace character for inputs for the scanf(), sscanf() and fscanf() functions. So, the sscanf() function will only copy upto the string “hello” into the array **s**. Hence, the output of the printf() function be only the string “hello”.

4. What will be the output of the following C code?

```
#include <stdio.h>
```

```
int main()
{
    short int i;
    scanf("%hd", &i);
    printf("%hd", i);
    return 0;
}
```

a) Compilation error
b) Undefined behavior
c) Whatever user types
d) None of the mentioned

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    short int i;
    scanf("%*d", &i);
    printf("%hd", i);
    return 0;
}
```

a) Compilation error
b) Somegarbage value
c) Whatever user types
d) Depends on the standard

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    short int i;
    scanf("%*hd", &i);
    printf("%hd", i);
    return 0;
}
```

a) Compilation error
b) Somegarbage value
c) Whatever user types
d) Depends on the standard

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    short int i;
    scanf("%h*d", &i);
    printf("%hd", i);
    return 0;
}
```

a) Compilation error
b) Undefined behavior
c) Somegarbage value
d) Depends on the standard.

ANSWER:- a
Explanation: None.

8. Which of the following is NOT a delimiter for an input in scanf?
a) Enter
b) Space
c) Tab
d) None of the mentioned

ANSWER:- d
Explanation: None.

9. If the conversion characters of int d, i, o, u and x are preceded by h, it indicates?
a) A pointer to int
b) A pointer to short
c) A pointer to long
d) A pointer to char

ANSWER:- b
Explanation: None.

1. Which of the following doesn't require an & for the input in scanf()?
a) char name[10];
b) int name[10];
c) float name[10];
d) all of the mentioned

ANSWER:- a
Explanation: None.

2. Which of the following is an invalid method for input?
a) scanf("%d%d%d",&a, &b, &c);
b) scanf("%d %d %d", &a, &b, &c);

c) scanf(“Three values are %d %d %d”,&a,&b,&c);
d) none of the mentioned

ANSWER:- d
Explanation: None.

3. Which of the following represents the function for scanf()?
a) void scanf(char *format, ...)
b) int scanf(char *format, ...)
c) char scanf(int format, ...)
d) char *scanf(char *format, ...)

ANSWER:- b
Explanation: None.

4. What does scanf() function return?
a) Number of successfully matched and assigned input items
b) Nothing
c) Number of characters properly printed
d) Error

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int n;
    scanf("%d", n);
    printf("%d", n);
}
```

a) Prints the number that was entered
b) Segmentation fault
c) Nothing
d) Varies

ANSWER:- c
Explanation: None.

6. What will be the output of the following C statement?

int sscanf(char *string, char *format, arg1, arg2, ...)

a) Scans the string according to the format in format and stores the resulting values through arg1, arg2, etc
b) The arguments arg1,arg2 etc must be pointers
c) Scans the string according to the format in format and stores the resulting values through arg1, arg2, etc, those arguments arg1,arg2 etc must be pointers
d) None of the mentioned

ANSWER:- c
Explanation: None.

7. The conversion characters d, i, o, u, and x may be preceded by h in scanf() to indicate?
a) A pointer to short
b) A pointer to long
c) Nothing
d) Error

ANSWER:- a
Explanation: None.

8. What will be the output of the following C code (when 4 and 5 are entered)?

```
#include <stdio.h>
void main()
{
    int m, n;
    printf("enter a number");
    scanf("%d", &n);
    scanf("%d", &m);
    printf("%d\t%d\n", n, m);
}
```

a) Error
b) 4 junkvalue
c) Junkvalue 5
d) 4 5

ANSWER:- d
Explanation: None.

1. What are the first and second arguments of fopen?
a) A character string containing the name of the file & the second argument is the mode
b) A character string containing the name of the user & the second argument is the mode
c) A character string containing file pointer & the second argument is the mode
d) None of the mentioned

ANSWER:- a
Explanation: None.

2. For binary files, a ___ must be appended to the mode string.
a) Nothing
b) “b”
c) “binary”

d) “01”

ANSWER:- b
Explanation: None.

Note: Join free Sanfoundry classes at Telegram or Youtube

3. What will fopen will return, if there is any error while opening a file?
a) Nothing
b) EOF
c) NULL
d) Depends on compiler

ANSWER:- c
Explanation: None.

4. What is the return value of getc()?
a) The next character from the stream is not referred by file pointer
b) EOF for end of file or error
c) Nothing
d) None of the mentioned

ANSWER:- b
Explanation: None.

Take C Programming Tests Now!

5. When a C program is started, O.S environment is responsible for opening file and providing pointer for that file?
a) Standard input
b) Standard output
c) Standard error
d) All of the mentioned

ANSWER:- d
Explanation: None.

6. In C language, FILE is of which data type?
a) int
b) char *
c) struct
d) None of the mentioned

ANSWER:- c
Explanation: None.

7. What is meant by ‘a’ in the following C operation?

```
fp = fopen("Random.txt", "a");
```

a) Attach
b) Append
c) Apprehend
d) Add

ANSWER:- b
Explanation: None.

8. Which of the following mode argument is used to truncate?
a) a
b) f
c) w
d) t

ANSWER:- c
Explanation: None.

9. Which type of files can’t be opened using fopen()?
a) .txt
b) .bin
c) .c
d) none of the mentioned

ANSWER:- d
Explanation: None.

1. Which of the following fopen() statements are illegal?
a) fp = fopen(“abc.txt”, “r”);
b) fp = fopen(“/home/user1/abc.txt”, “w”);
c) fp = fopen(“abc”, “w”);
d) none of the mentioned

ANSWER:- d
Explanation: None.

2. What does the following segment of C code do?

```
fprintf(fp, "Copying!");
```

a) It writes “Copying!” into the file pointed by fp
b) It reads “Copying!” from the file and prints on display
c) It writes as well as reads “Copying!” to and from the file and prints it
d) None of the mentioned

ANSWER:- a
Explanation: None.

3. What is FILE reserved word?
a) A structure tag declared in stdio.h
b) One of the basic data types in c
c) Pointer to the structure defined in stdio.h
d) It is a type name defined in stdio.h

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    FILE *fp = stdin;
    int n;
    fprintf(fp, "%d", 45);
}
```

a) Compilation error
b) 45
c) Nothing
d) Depends on the standard

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    FILE *fp = stdout;
    int n;
    fprintf(fp, "%d", 45);
}
```

a) Compilation error
b) 45
c) Nothing
d) Depends on the standard

ANSWER:- b
Explanation: None.

6. stdout, stdin and stderr are _____
a) File pointers
b) File descriptors
c) Streams
d) Structure

ANSWER:- a
Explanation: None.

7. Which of the following statements about stdout and stderr are true?
a) Same
b) Both connected to screen always
c) Both connected to screen by default
d) stdout is line buffered but stderr is unbuffered

ANSWER:- c
Explanation: None.

8. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    FILE *fp = stdout;
    int n;
    fprintf(fp, "%d ", 45);
    fprintf(stderr, "%d ", 65);
    return 0;
}
```

a) 45 65
b) 65 45
c) 65
d) Compilation error

ANSWER:- b
Explanation: None.

9. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    FILE *fp = stdout;
    int n;
    fprintf(fp, "%d\n ", 45);
    fprintf(stderr, "%d ", 65);
    return 0;
}
```

a) 45 65
b) 65 45
c) 65
d) Compilation error

ANSWER:- a
Explanation: None.

10. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
```

```
        FILE *fp = stdout;
        int n;
        fprintf(fp, "%d ", 45);
        fflush(stdout);
        fprintf(stderr, "%d",
    65);
        return 0;
    }
```

a) 45 65
b) 65 45
c) 45
d) Compilation error

ANSWER:- a
Explanation: None.

1. What is the output of the following C code if there is no error in stream fp?

```
#include <stdio.h>
int main()
{
    FILE *fp;
    fp = fopen("newfile",
"w");
    printf("%d\n",
ferror(fp));
    return 0;
}
```

a) Compilation error
b) 0
c) 1
d) Any nonzero value

ANSWER:- b
Explanation: None.

2. Within main, return expr statement is equivalent to _____
a) abort(expr)
b) exit(expr)
c) ferror(expr)
d) none of the mentioned

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    FILE *fp;
    char c;
    int n = 0;
    fp =
fopen("newfile1.txt", "r");
    while (!feof(fp))
    {
        c = getc(fp);
        putc(c, stdout);
    }
}
```

a) Compilation error
b) Prints to the screen content of newfile1.txt completely
c) Prints to the screen some contents of newfile1.txt
d) None of the mentioned

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    FILE *fp = stdout;
    stderr = fp;
    fprintf(stderr, "%s",
"hello");
}
```

a) Compilation error
b) hello
c) Undefined behaviour
d) Depends on the standard

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char buf[12];
    stderr = stdin;
    fscanf(stderr, "%s",
buf);
```

```
7.          printf("%s\n", buf);
8.      }
```

a) Compilation error
b) Undefined behaviour
c) Whatever user types
d) None of the mentioned

ANSWER:- c
Explanation: None.

6. stderr is similar to?
a) stdin
b) stdout
c) both stdout and stdin
d) none of the mentioned

ANSWER:- a
Explanation: None.

7. What happens when we use the following C statement?

```
fprintf(stderr, "error: could not open filen");
```

a) The diagnostic output is directly displayed in the output
b) The diagnostic output is pipelined to the output file
c) The line which caused error is compiled again
d) The program is immediately aborted

ANSWER:- a
Explanation: None.

8. Which of the following function can be used to terminate the main function from another function safely?
a) return(expr);
b) exit(expr);
c) abort();
d) both exit(expr); and abort();

ANSWER:- b
Explanation: None.

1. Which of the following causes an error?
a) Trying to read a file that doesn't exist
b) Inability to write data in a file
c) Failure to allocate memory with the help of malloc
d) All of the mentioned

ANSWER:- d
Explanation: None.

2. What is the purpose of the C function?


```
int ferror(FILE *fp)
```

a) They check for input errors
b) They check for output errors
c) They check for all types of errors
d) They check for error in accessing the file

ANSWER:- b
Explanation: None.

3. stderr is similar to?
a) stdin
b) stdout
c) Both stdout and stdin
d) None of the mentioned

ANSWER:- b
Explanation: stderr is not exactly the same as stdout, but similar in the sense that both puts the output or error to the monitor.

4. What will be the output of the following C statement?

```
fprintf(stderr, "error: could not open filen");
```

a) The diagnostic output is directly displayed in the output
b) The diagnostic output is pipelined to the output file
c) The line which caused error is compiled again
d) The program is immediately aborted

ANSWER:- a
Explanation: None.

5. Which of the following function can be used to terminate the main() function from another function safely?
a) return(expr);
b) exit(expr);
c) abort();
d) both exit(expr); and abort();

ANSWER:- b
Explanation: None.

6. Which of the following causes an error?
a) Trying to read a file that doesn't exist
b) Inability to write data in a file
c) Failure to allocate memory with the help of malloc
d) All of the mentioned

ANSWER:- d
Explanation: None.

7. What is the purpose of the C function?

```
int ferror(FILE *fp)
```

a) They check for input errors
b) They check for output errors
c) They check for all types of errors
d) They check for error in accessing the file

ANSWER:- b
Explanation: None.

1. The syntax of fgets is char *fgets(char *line, int maxline, FILE *fp). Which is true for fgets?
a) Returns line on success
b) On end of file or error it returns NULL
c) Nothing
d) Both returns line on success & On end of file or error it returns NULL

ANSWER:- d
Explanation: None.

2. fputs() function writes a string to a file that only ends with a newline.
a) True
b) False
c) Depends on the standard
d) Depends on the compiler

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.      char line[3];
6.      fgets(line, 3, stdin);
7.      printf("%d\n",
    strlen(line));
8.      return 0;
9.  }
```

a) 3
b) 1
c) Any length since line did not end with null character
d) Depends on the standard

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.      char line[3];
6.      FILE *fp;
7.      fp = fopen("newfile.txt",
    "r");
8.      while (fgets(line, 3,
    fp))
9.      fputs(line, stdout);
10.      return 0;
11.  }
```

a) Compilation error
b) Infinite loop
c) Segmentation fault
d) No.of lines present in file newfile

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code if 2 characters is typed by the user?

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.      char line[3];
6.      fgets(line, 3, stdin);
7.      printf("%d\n", line[2]);
8.      return 0;
9.  }
```

a) Compilation error
b) Undefined behaviour
c) 0
d) 10(ascii value of newline character)

ANSWER:- c
Explanation: None.

6. fputs() adds newline character.
a) True
b) False
c) Depends on the standard

d) Undefined behaviour

ANSWER:- b
Explanation: None.

7. puts() function adds newline character.
a) True
b) False
c) Depends on the standard
d) Undefined behaviour

ANSWER:- a
Explanation: None.

8. gets() function checks overflow run.
a) True
b) False
c) Depends on the standard
d) Undefined behaviour

ANSWER:- b
Explanation: None.

9. puts() does the following when it writes to stdout.
a) Deletes everything
b) Adds 't' to the line written
c) Deletes the terminating 'n'
d) Adds 'n' to the line written

ANSWER:- d
Explanation: None.

1. What is the size of array "line" used in fgets(line, maxline, *fp) function?
a) maxline – 1
b) maxline
c) maxline + 1
d) Size is dynamic

ANSWER:- b
Explanation: None.

2. What will be the output of the following C function when EOF returns?

```
int fputs(char *line, FILE *fp)
```

a) '' character of array line is encountered
b) 'n' character in array line is encountered
c) 't' character in array line is encountered
d) When an error occurs

ANSWER:- d
Explanation: None.

3. Identify X library function for line input and output in the following C code?

```
#include <stdio.h>
int X(char *s, FILE *iop)
{
    int c;
    while (c = *s++)
    putc(c, iop);
    return ferror(iop) ?
EOF : 0;
}
```

a) getc
b) putc
c) fgets
d) fputs

ANSWER:- d
Explanation: None.

4. Which function has a return type as char pointer?
a) getline
b) fputs
c) fgets
d) all of the mentioned

ANSWER:- c
Explanation: None.

5. Which of the following is the right declaration for fgets() inside the library?
a) int *fgets(char *line, int maxline, FILE *fp);
b) char *fgets(char *line, int maxline, FILE *fp);
c) char *fgets(char *line, FILE *fp);
d) int *fgets(char *line, FILE *fp);

ANSWER:- b
Explanation: None.

6. what is the return value of fputs()?
a) EOF if an error occurs
b) Non-negative if no error
c) EOF if an error occurs & Non-negative if no error
d) None of the mentioned

ANSWER:- c
Explanation: None.

7. gets() and puts() operate on _____
a) stdin and stdout
b) files
c) stderr
d) nothing

ANSWER:- a
Explanation: None.

8. gets() does the following when it reads from stdin.
a) Deletes the 't'
b) Puts adds it.
c) Deletes the terminating 'n'
d) Nothing

ANSWER:- c
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello,
    world";
5.        char *str1 = "hello,
    world";
6.        if (strcmp(str, str1))
7.            printf("equal");
8.        else
9.            printf("unequal");
10.    }
```

a) equal
b) unequal
c) Compilation error
d) Depends on the compiler

ANSWER:- b
Explanation: None.

2. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello,
    world";
5.        char str1[15] = "hello
    wo 9";
6.        strcpy(str, str1);
7.        printf("%s", str1);
8.    }
```

a) Compilation error
b) Segmentation Fault
c) hello, world
d) hello, wo 9

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #include <string.h>
3.    int main()
4.    {
5.        char *str = "hello,
    world";
6.        char str1[9];
7.        strncpy(str1, str, 9);
8.        printf("%s %d", str1,
    strlen(str1));
9.    }
```

a) hello, world 11
b) hello, wor 9
c) Undefined behaviour
d) Compilation error

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char *str = "hello,
    world\n";
5.        printf("%d",
    strlen(str));
6.
7.    }
```

a) Compilation error
b) Undefined behaviour
c) 13
d) 11

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    int main()
3.    {
4.        char str[11] = "hello";
```

```
5.          char *str1 = "world";
6.          strcat(str, str1);
7.          printf("%s %d", str,
    str[10]);
8.      }
```

a) helloworld 0
b) helloworld anyvalue
c) worldhello 0
d) Segmentation fault/code crash

ANSWER:- a
Explanation: None.

6. Strcat() function adds null character.
a) Only if there is space
b) Always
c) Depends on the standard
d) Depends on the compiler

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          char str[10] = "hello";
5.          char *str1 = "world";
6.          strncat(str, str1, 9);
7.          printf("%s", str);
8.      }
```

a) helloworld
b) Undefined behaviour
c) helloworl
d) hellowor

ANSWER:- a
Explanation: None.

1. What type of return-type used in String operations?
a) void only
b) void and (char *) only
c) void and int only
d) void, int and (char *) only

ANSWER:- d
Explanation: None.

2. String operation such as strcat(s, t), strcmp(s, t), strcpy(s, t) and strlen(s) heavily rely upon.
a) Presence of NULL character
b) Presence of new-line character
c) Presence of any escape sequence
d) None of the mentioned

ANSWER:- a
Explanation: None.

3. Which pre-defined function returns a pointer to the last occurence of a character in a string?
a) strchr(s, c);
b) strrchr(s, c);
c) strlchr(s, c);
d) strfchr(s, c);

ANSWER:- b
Explanation: None.

4. Which of the following function compares 2 strings with case-insensitively?
a) strcmp(s, t)
b) strcmpcase(s, t)
c) strcasecmp(s, t)
d) strchr(s, t)

ANSWER:- c
Explanation: None.

5. What will be the value of var for the following C statement?

```
var = strcmp("Hello", "World");
```

a) -1
b) 0
c) 1
d) strcmp has void return-type

ANSWER:- a
Explanation: None.

6. What will be the output of the following C code?

```
1.      #include <stdio.h>
2.      int main()
3.      {
4.          char str[10] = "hello";
5.          char *p = strrchr(str,
    '1');
6.          printf("%c\n", *(++p));
7.      }
```

a) l
b) o
c) e
d) Compilation error

ANSWER:- b
Explanation: None.

1. Which of the following library function is not case-sensitive?
a) toupper()
b) tolower()
c) isdigit()
d) all of the mentioned

ANSWER:- c
Explanation: None.

2. The following C expression can be substituted for?

```
if (isalpha(c) && isdigit(c))
```

a) if (isalnum(c))
b) if (isalphanum(c))
c) if (isalphanumeric(c))
d) none of the mentioned

ANSWER:- d
Explanation: None.

Check this: BCA MCQs | Advanced C Programming Videos

3. Which of the following will return a non-zero value when checked with isspace(c)?
a) blank
b) newline
c) return
d) all of the mentioned

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    char i = 9;
    if (isdigit(i))
        printf("digit\n");
    else
        printf("not digit\n");
     return 0;
}
```

a) digit
b) not digit
c) Depends on the compiler
d) None of the mentioned

ANSWER:- b
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    int i = 9;
    if (isdigit(i))
        printf("digit\n");
    else
        printf("not digit\n");
     return 0;
}
```

a) digit
b) not digit
c) Depends on the compiler
d) None of the mentioned

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char i = '9';
    if (isdigit(i))
        printf("digit\n");
    else
        printf("not digit\n");
     return 0;
}
```

a) digit
b) not digit
c) Depends on the compiler
d) None of the mentioned

ANSWER:- a
Explanation: None.

7. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    int i = 0;
    if (isspace(i))
        printf("space\n");
    else
        printf("not space\n");
     return 0;
}
```

a) Compile time error
b) space
c) not space
d) None of the mentioned

ANSWER:- c
Explanation: The value of variable i is 0 which is the NULL character in ASCII. Hence, the output will be printed as “not space”.

8. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    int i = 32;
    if (isspace(i))
        printf("space\n");
    else
        printf("not space\n");
     return 0;
}
```

a) Compile time error
b) space
c) not space
d) None of the mentioned

ANSWER:- b
Explanation: The ASCII value of space character is 32. Since the variable i stores 32, the output will be printed as “space”.

1. Which is true about isaplpha(c), where c is an int that can be represented as an unsigned?

```
char or EOF.isalpha(c) returns
```

a) Non-zero if c is alphabetic
b) 0 if c is not alphabetic
c) Both Non-zero if c is alphabetic & 0 if c is not alphabetic
d) None of the mentioned

ANSWER:- c
Explanation: None.

2. Which is true about isupper(c), where c is an int that can be represented as an unsigned?

```
char or EOF.isupper(c) returns
```

a) Non-zero if c is upper case
b) 0 if c is not upper case
c) Nothing
d) Both Non-zero if c is upper case & 0 if c is not upper case

ANSWER:- d
Explanation: None.

3. Which is true about isalnum(c), where c is an int that can be represented as an unsigned?

```
char or EOF.isalnum(c) returns
```

a) Non-zero if isalpha(c) or isdigit(c)
b) 0 if not isalpha(c) or not isdigit(c)
c) Both Non-zero if isalpha(c) or isdigit(c) & 0 if not isalpha(c) or not isdigit(c)
d) None of the mentioned

ANSWER:- c
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    char c = 't';
    printf("%d\n", isspace(c));
}
```

a) Non-zero number
b) Nothing
c) Error
d) t

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
#include <ctype.h>
int main()
{
    char c = 't';
    printf("is :%c\n",
tolower('A'));
}
```

a) A
b) a
c) Non-zero number
d) Zero

ANSWER:- b
Explanation: None.

6. Which types of input are accepted in toupper(c)?
a) char
b) char *
c) float
d) Both char and char *

ANSWER:- a
Explanation: None.

7. What is the difference in the ASCII value of capital and non-capital of the same letter is?
a) 1
b) 16
c) 32
d) Depends with compiler

ANSWER:- c
Explanation: None.

1. ungetc() can be used only with getc().
a) true
b) false
c) depends on the standard
d) depends on the platform

ANSWER:- b
Explanation: None.

2. Which character of pushback is guaranteed per file?
a) True
b) False
c) Depends on the compiler
d) Depends on the platform

ANSWER:- a
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", &n);
    ungetc(n, stdin);
    scanf("%d", &n);
    printf("%d\n", n);
    return 0;
 }
```

a) Compile time error
b) Whatever is typed by the user first time
c) Whatever is typed by the user second time
d) Undefined behaviour

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char n[20];
    fgets(n, 19, stdin);
    ungetc(n[0], stdin);
    scanf("%s", n);
    printf("%s\n", n);
    return 0;
 }
```

a) Compile time error
b) Whatever string user types second time
c) Whatever string user types first time
d) First character of whatever user types first time and whatever user types second time

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code considering user typed jkl?

```
#include <stdio.h>
```

```
2.     int main()
3.     {
4.         char n[20];
5.         fgets(n, 19, stdin);
6.         ungetc(n[0], stdin);
7.         printf("%s\n", n);
8.         return 0;
9.     }
```

a) jkl
b) kl
c) Undefined behaviour
d) jk

ANSWER:- a
Explanation: None.

6. How many characters for pushback is guaranteed per file while using ungetc(c, fp);?
a) Only 1 character
b) Characters within 1 word
c) Characters within 1st new-line
d) All characters upto NULL character

ANSWER:- a
Explanation: None.

7. Which of the following is the correct syntax for calling function ungetc?

```
Assume int c and FILE *fp
```

a) ungetc(c,*fp);
b) ungetc(c, fp);
c) ungetc(fp, c);
d) ungetc(*fp,c);

ANSWER:- b
Explanation: None.

8. ungetc() is used _____
a) to get a char
b) to get an int
c) to push a character back to file
d) nothing

ANSWER:- c
Explanation: None.

1. Which of the following is the correct declaration for ungetc?
a) int ungetc(int c, FILE fp);
b) int ungetc(int *c, FILE fp);
c) int ungetc(int c, FILE *fp);
d) int ungetc(int *c, FILE *fp);

ANSWER:- c
Explanation: None.

2. Which of the following cannot be used with ungetc()?
a) scanf
b) getc
c) getchar
d) printf

ANSWER:- d
Explanation: None.

3. What does the ungetc function return for the following C expression?

```
ungetc(c, fp);//where declarations are int c and FILE *fp
```

a) It returns character c
b) It returns EOF for an error
c) Both returns character c and returns EOF for an error
d) Either returns character c or returns EOF for an error

ANSWER:- d
Explanation: None.

4. What will be the output of the following C statement?

```
int ungetc(int c, FILE *fp)
```

a) Either c or EOF for an error
b) Nothing
c) fp
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. Only ____character of pushback is guaranteed per file when ungetc is used.
a) Two
b) One
c) Many
d) Zero

ANSWER:- b
Explanation: None.

6. ungetc() may be used with _____
a) scanf
b) getc
c) getchar

d) all of the mentioned

ANSWER:- d
Explanation: None.

7. What is the syntax of ungetc()?
a) void ungetc(int c, FILE *fp)
b) int ungetc(int c, FILE *fp)
c) int ungetc(String c, FILE *fp)
d) int getc(int c, FILE *fp)

ANSWER:- b
Explanation: None.

1. The function ____ obtains a block of memory dynamically.
a) calloc
b) malloc
c) both calloc & malloc
d) free

ANSWER:- c
Explanation: None.

2. void * malloc(size_t n) returns?
a) Pointer to n bytes of uninitialized storage
b) NULL if the request can be satisfied
c) Nothing
d) None of the mentioned

ANSWER:- a
Explanation: None.

3. calloc() returns storage that is initialized to.
a) Zero
b) Null
c) Nothing
d) One

ANSWER:- a
Explanation: None.

4. In function free(p), p is a _____
a) int
b) pointer returned by malloc()
c) pointer returned by calloc()
d) pointer returned by malloc() & calloc()

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    void main()
3.    {
4.        char *p = calloc(100, 1);
5.        p = "welcome";
6.        printf("%s\n", p);
7.    }
```

a) Segmentation fault
b) Garbage
c) Error
d) welcome

ANSWER:- d
Explanation: None.

6. Memory allocation using malloc() is done in _____
a) Static area
b) Stack area
c) Heap area
d) Both Stack & Heap area

ANSWER:- c
Explanation: None.

7. Why do we write (int *) before malloc?

```
int *ip = (int *)malloc(sizeof(int));
```

a) It is for the syntax correctness
b) It is for the type-casting
c) It is to inform malloc function about the data-type expected
d) None of the mentioned

ANSWER:- b
Explanation: None.

8. Which of the following is used during memory deallocation in C?
a) remove(p);
b) delete(p);
c) free(p);
d) terminate(p);

ANSWER:- c
Explanation: None.

1. Which of the following will return a result most quickly for searching a given key?
a) Unsorted Array
b) Sorted Array
c) Sorted linked list
d) Binary Search Tree

ANSWER:- d
Explanation: None.

2. On freeing a dynamic memory, if the pointer value is not modified, then the pointer points to.
a) NULL

b) Other dynamically allocated memory
c) The same deallocated memory location
d) It points back to the location it was initialized with

ANSWER:- c
Explanation: None.

3. Which of the following should be used for freeing the memory allocated in the following C code?

```
#include <stdio.h>
struct p
{
    struct p *next;
    int x;
};
int main()
{
    struct p *p1 = (struct p*)malloc(sizeof(struct p));
    p1->x = 1;
    p1->next = (struct p*)malloc(sizeof(struct p));
    return 0;
}
```

a)

```
free(p1);
free(p1->next)
```

b)

```
free(p1->next);
free(p1);
```

c) free(p1);
d) all of the mentioned

ANSWER:- b
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    struct p *next;
    int x;
};
int main()
{
    struct p *p1 = calloc(1, sizeof(struct p));
    p1->x = 1;
    p1->next = calloc(1, sizeof(struct p));
    printf("%d\n", p1->next->x);
    return 0;
}
```

a) Compile time error
b) 1
c) Somegarbage value
d) 0

ANSWER:- d
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
struct p
{
    struct p *next;
    int x;
};
int main()
{
    struct p* p1 = malloc(sizeof(struct p));
    p1->x = 1;
    p1->next = malloc(sizeof(struct p));
    printf("%d\n", p1->next->x);
    return 0;
}
```

a) Compile time error
b) 1
c) Somegarbage value
d) 0

ANSWER:- c
Explanation: None.

6. calloc() initialize memory with all bits set to zero.

a) True
b) False
c) Depends on the compiler
d) Depends on the standard

ANSWER:- a
Explanation: None.

7. What if size is zero in the following C statement?

```
realloc(ptr, size)
```

a) Allocate a memory location with zero length
b) Free the memory pointed to by ptr
c) Undefined behaviour
d) Doesn't do any reallocation of ptr i.e. no operation

ANSWER:- b
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
#include <math.h>
int main()
{
    int i = 90;
    printf("%f\n", sin(i));
    return 0;
}
```

a) Compile time error
b) Undefined behaviour
c) 0.893997
d) 1.000000

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
#include <math.h>
int main()
{
    unsigned int i = -1;
    printf("%f\n", fabs(i));
    return 0;
}
```

a) Compile time error
b) 1
c) -1
d) None of the mentioned

ANSWER:- d
Explanation: None.

3. function fabs defined math.h header file takes the argument of type integer.
a) True
b) False
c) Depends on the implementation
d) Depends on the standard

ANSWER:- b
Explanation: None.

4. log(x) function defined in math.h header file is _____
a) Natural base logarithm
b) Logarithm to the base 2
c) Logarithm to the base 10
d) None of the mentioned

ANSWER:- a
Explanation: None.

5. What will be the output of the following C code?

```
#include <stdio.h>
#include <math.h>
int main()
{
    int i = 10;
    printf("%f\n", log10(i));
    return 0;
}
```

a) Compile time error
b) 1.000000
c) 2.302585
d) None of the mentioned

ANSWER:- b
Explanation: None.

6. What type of inputs are accepted by mathematical functions?
a) short
b) int
c) float
d) double

ANSWER:- d
Explanation: None.

7. In linux, apart from including math header file, the program is successfully executed by which of the following?
a) cc filename.c
b) cc filename.c -lc
c) cc -math filename.c
d) cc -lm filename.c

ANSWER:- d
Explanation: None.

8. Which of the following is not a valid mathematical function?
a) frexp(x);
b) atan2(x,y);
c) srand(x);
d) fmod(x);

ANSWER:- d
Explanation: None. 1. What is function srand(unsigned)?
a) Sets the seed for rand
b) Doesn't exist
c) Is an error
d) None of the mentioned

ANSWER:- a
Explanation: None.

2. Which is the best way to generate numbers between 0 to 99?
a) rand()-100
b) rand()%100
c) rand(100)
d) srand(100)

ANSWER:- b
Explanation: None.

3. Which is the correct way to generate numbers between minimum and maximum(inclusive)?
a) minimum + (rand() % (maximum – minimum));
b) minimum + (rand() % (maximum – minimum + 1));
c) minimum * (rand() % (maximum – minimum))
d) minimum – (rand() % (maximum + minimum));

ANSWER:- b
Explanation: None.

4. rand() and srand() functions are used _____
a) To find sqrt
b) For and operations
c) For or operations
d) To generate random numbers

ANSWER:- d
Explanation: None.

5. What is the return type of rand() function?
a) short
b) int
c) long
d) double

ANSWER:- b
Explanation: None.

6. Which of the following can be used for random number generation?
a) random()
b) rnd()
c) rndm()
d) none of the mentioned

ANSWER:- a
Explanation: None.

7. Which of the following snippet will effectively generate random numbers?
a) rand();
b) rand(10);
c) rand(time(NULL));
d) all of the mentioned

ANSWER:- a
Explanation: None.

8. Which among the following is correct function call for rand() and random()?
a) rand() and random();
b) rand() and random(1);
c) rand(1) and random(1);
d) rand(1) and random();

ANSWER:- a
Explanation: None.

9. For the function call time(), what type of parameter is accepted?
a) int
b) int *
c) time_t
d) time_t *

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
1.    #include <stdio.h>
2.    #include <stdlib.h>
3.    int main()
```

```
{
    printf("%d\n", rand() % 1000);
    return 0;
}
```

a) Compile time error
b) An integer between 0-1000
c) An integer between 0-999 including 0 and 999
d) An integer between 0-1000 including 1000

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    srand(9000);
    printf("%d\n", rand());
    return 0;
}
```

a) Compile time error
b) An integer in the range 0 to RAND_MAX
c) A double in the range 0 to 1
d) A float in the range 0 to 1

ANSWER:- b
Explanation: None.

3. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    printf("%d\n", srand(9000));
    return 0;
}
```

a) Compile time error
b) An integer in the range 0 to 9000
c) A float in the range 0 to 1
d) A double in the range 0 to 9000

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    srand(time(NULL));
    printf("%d\n", rand());
    return 0;
}
```

a) Compile time error
b) An integer in the range 0 to RAND_MAX
c) A double in the range 0 to 1
d) A float in the range 0 to 1

ANSWER:- b
Explanation: None.

5. In the below C program, every time program is run different numbers are generated.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    printf("%d\n", rand());
    return 0;
}
```

a) True
b) False
c) Depends on the platform
d) Depends on the compiler

ANSWER:- b
Explanation: None.

6. In the following C program, every time program is run different numbers are generated.

```
#include <stdio.h>
int main()
{
    srand(time(NULL));
    printf("%d\n", rand());
    return 0;
}
```

a) True
b) False
c) Depends on the platform
d) Depends on the compiler

ANSWER:- a
Explanation: None.

7. Which of these is a correct way to generate numbers between 0 to 1(inclusive) randomly?
a) rand() / RAND_MAX
b) rand() % 2
c) rand(0, 1)
d) none of the mentioned

ANSWER:- a
Explanation: None.

1. Which of the following mathematical function requires 2 parameter for successful function call?
a) fmod();
b) div();
c) atan2();
d) all of the mentioned

ANSWER:- d
Explanation: None.

2. Which mathematical function among the following does NOT require int parameters?
a) div(x, y);
b) srand(x);
c) sqrt(x);
d) all of the mentioned

ANSWER:- c
Explanation: None.

3. What will sin(x) returns?
a) sine of x where x is in radians
b) sine of x where x is in degree
c) cosine of x where x is in radians
d) cosine of x where x is in degree

ANSWER:- a
Explanation: None.

4. What will cos(x) return?
a) sine of x where x is in radians
b) sine of x where x is in degree
c) cosine of x where x is in radians
d) cosine of x where x is in degree

ANSWER:- c
Explanation: None.

5. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  #include <math.h>
3.  void main()
4.  {
5.      int k = pow(2, 3);
6.      printf("%d\n", k);
7.  }
```

a) 9
b) 8
c) -1
d) 6

ANSWER:- b
Explanation: None.

6. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  #include <math.h>
3.  void main()
4.  {
5.      int k = fabs(-87);
6.      printf("%d\n", k);
7.  }
```

a) -87
b) 87
c) 78
d) error

ANSWER:- b
Explanation: None.

7. What will be the output of the following C code?

```
1.  #include <stdio.h>
2.  #include <math.h>
3.  void main()
4.  {
5.      int k = sqrt(-4);
6.      printf("%d\n", k);
7.  }
```

a) -2
b) 2
c) Compile time error
d) NaN

ANSWER:- d
Explanation: None.

8. Which among the following mathematical function do not have a “double” return-type?
a) srand(x);
b) ceil(x);
c) floor(x);
d) both ceil(x); and floor(x);

ANSWER:- a
Explanation: None.

1. The syntax of printf() function is printf("control string", variable list) ;what is the prototype of the control string?
a) %[flags][.precision][width][length]specifier
b) %[flags][length][width][.precision]specifier
c) %[flags][width][.precision][length]specifier
d) %[flags][.precision][length][width]specifier

ANSWER:- c
Explanation: The prototype of control string is %[flags][width][.precision][length]specifier. Each control string must begin with % sign.

2. The parameter control string in the printf () is a C String that contains text to be _____
a) taken from a standard output device
b) written on to the standard output device
c) received from the standard output device
d) nothing can be said

ANSWER:- b
Explanation: After the control string, the function can have many additional arguments as specified in the control string, this parameter contains the text to be written on to the standard output device.

3. Output justification such as decimal point, numerical sign, trailing zeros or octal are specified.
a) specifier
b) flags
c) precision
d) decimal

ANSWER:- b
Explanation: Flags specify output justification such as Left-justify within the data given field width, Displays the data with its numeric sign, used to provide additional specifiers like o, x, X for octal, left padding of a number.

4. What symbol is used to Left-justify within the data given field width?
a) -(minus sign)
b) +(plus sign)
c) #
d) 0

ANSWER:- a
Explanation: To left-justify the data use minus sign(-) in the flags field.

5. What specifies the minimum number of characters to print after being padded with zeros or blank spaces?
a) flags
b) length
c) width
d) precision

ANSWER:- c
Explanation: width specifies the minimum number of positions in the output.

6. The maximum number of characters to be printed is specified by _____
a) precision
b) width
c) length
d) flags

ANSWER:- a
Explanation: Precision specifies the maximum number of characters to print.

7. _____is used to define the type and the interpretation of the value of the corresponding argument.
a) precision
b) specifiers
c) flags
d) decimal

ANSWER:- b
Explanation: Specifiers is used to define the type and the interpretation of the value of the corresponding argument. Example: c for a single character, d for decimal values etc.

8. A conversion specification %7.4f means _____
a) print a floating point value of maximum 7 digits where 4 digits are allotted for the digits after the decimal point
b) print a floating point value of maximum 4 digits where 7digits are allotted for the digits after the decimal point
c) print a floating point value of maximum 7 digits
d) print a floating point value of minimum 7 digits where 4 digits are allotted for the digits after the decimal point

ANSWER:- a
Explanation: The conversion specification %7.4f means that it will print floating point number maximum of 7 digits and 4 digits after the decimal point.

9. Choose the correct description for control string %-+7.2f.
a) – means display the sign, + means left justify, 7 specifies the width and 2 specifies the precision

b) – means left justify, + means display the sign, 7 specifies the width and 2 specifies the precision
c) – means display the sign, + means left justify, 7 specifies the precision and 2 specifies the width
d) – means left justify, + means display the sign, 7 specifies the precision and 2 specifies the width

ANSWER:- b
Explanation: The given control string %-+7.2f means that – is for left justify, + to display sign, 7 specifies the precision and 2 specifies the width.

10. What error is generated on placing an address operator with a variable in the printf statement?
a) compile error
b) run-time error
c) logical error
d) no error

ANSWER:- b
Explanation: Placing an address operator with a variable in the printf statement will generate a run-time error.

1. If by mistake you specify more number of arguments, the excess arguments will _____
a) be ignored
b) produce compile error
c) produce run-time error
d) produce logical error

ANSWER:- a
Explanation: The excess arguments will simply be ignored.

2. What happens when zero flag is used with left justification?
a) data is padded with zeros
b) zero flag is ignored
c) data is padded with blank spaces
d) will give error

ANSWER:- b
Explanation: Zero flag is not considered when used with left justification because adding zeros after a number changes its value.

3. For floating point numbers, the precision flag specifies the number of decimal places to be printed. When no precision modifier is specified, printf() prints _____
a) six decimal positions
b) five decimal positions
c) four decimal positions
d) three decimal positions

ANSWER:- a
Explanation: Its format can be given as “. m”, where m specifies the number of decimal digits when no precision modifier is specified, printf prints six decimal positions.

4. What will the given code result in printf(“\n you are\”awesome \" ");?
a) compile error
b) run-time error
c) you are "awesome"
d) you are awesome

ANSWER:- c
Explanation: The above given code uses \”<word>\” to display the word within double inverted commas on standard output screen.

5. What will be the output for the given code printf(“\n The number is %07d”,1212);
a) The number is 0001212
b) The number is 1212
c) The number is 1212
d) The number is 1212000

ANSWER:- a
Explanation: 0 in the above code is Flags. The number is left-padded with zeros(0) instead of spaces.

6. What will be the output of the following code?

```
char t=’N’;
printf(“\n %c \n %3c \n %5c”,t,t,t);
```

a)
```
N
  N
    N
```
b)
```
 N
   N
     N
```
c)
```
N
N
N
```
d) N N N

ANSWER:- b
Explanation: In the given code each argument is printed on a new line due to control character \n. Width mentioned in the above code is 1,3,5 hence the character is printed on a new line after being padded with blank spaces.

7. Select the right explanation to the given code.

```
printf(“%*. *f”, 5,4,5700);
```

a) the minimum field width has to be 4, the precision is given to be 5, and the value to be displayed is 5700
b) the minimum field width is 5, the precision is 4, and the value to be displayed is 5700
c) compile error
d) run-time error

ANSWER:- b
Explanation: The minimum field width and precision specifiers are usually constants. They can also be provided by arguments to printf(). This is done by using * modifier as shown in the given code.

8. What will be the output of the following C code?

```
char str[] = &quot;Hello Nancy“;
printf(“\n %.7s”, str) ;
```

a) Hello Nan
b) Hello
c) Hello N
d) Hello Nancy

ANSWER:- c
Explanation: The output for the code must be 7 characters including white spaces.

9. What will be the output of the following C code?

```
char str[] =”Too Good”;
printf(“\n %7s”,str);
```

a) Too Good
b) Too G
c) Too Go
d) Too

ANSWER:- a
Explanation: The complete string “Too Good” is printed. This is because if data needs more space than specified, then printf overrides the width specified by the user.

10. What will be the output of the following C code?

```
printf(“\n Output: %5d \t %x \t %#x”,
234,234,234);
```

a) Output:234EA0xEA
b) Output:00234 EA 0xEA
c) Output: 234 EA 0xEA
d) ERROR

ANSWER:- c
Explanation: The control character \t is used to provide gap between the words. %5d – the width of the string is set to 5, characters are printed after being padded with blank spaces.%x, %#x is additional specifiers for octal and hexadecimal values.

1. The syntax of the scanf() is scanf(“control string “, arg1,arg2,arg3,….,argn); the prototype of control string is _____
a) [=%[width][modifiers]type=]
b) [=%[modifiers][width]type=]
c) [=%[width] [modifiers]]
d) [width][modifiers]

ANSWER:- a
Explanation: scanf() starts with the symbol % followed by the width, modifier, type of the argument.

2. What is the use of symbol * in the control string as shown [=%[*][width] [modifiers] type=]?
a) * is optional and used when the data should be read from the stream but ignored
b) * is not optional, used to read data from the stream but it is not ignored
c) * is not optional, it is used to read data stream but ignored
d) * is optional and used to read data from stream but it is not ignored

ANSWER:- a
Explanation: * is an optional argument, it indicates that data should be read from the steam but ignored (not stored in a memory location)

3. What action is carried out by scanf if a user enters any blank spaces, tabs, and newlines?
a) consider as input
b) ignores it
c) produces error
d) nothing can be said

ANSWER:- b
Explanation: The scanf() function ignores any blank spaces, tabs, and newlines entered by the user. This scanf() function just returns the number of input fields successfully scanned and stored.

4. What error will generate if the read and write parameters are not separated by commas?
a) run-time error
b) compile error
c) logical error

d) no error

ANSWER:- b
Explanation: A compile error will be generated if the read and write parameters are not separated by commas.

5. What will be the output of the following C code?

```
char str[] =”Good”;
scanf(“%s”, str);
```

a) compile error
b) run-time error
c) good
d) logical error

ANSWER:- c
Explanation: String can be read from the stream without the use of address of operator (&).

6. What will be the output of the following C code?

```
scanf(“ %d %d %d”,&amp;n1,&amp;n2);
```

a) read data for two
b) generate error
c) read data for three
d) nothing can be said

ANSWER:- b
Explanation: The following scanf() statement will generate an error as no variable address is given for the third conversion specification.

7. What form the data must be entered for the given C code?

```
scanf(“%d / %d”, &amp;n1,&amp;n2);
```

a) 6 9
b) 6/9
c) compile error
d) run-time error

ANSWER:- b
Explanation: The slash in the control String are neither white space characters nor a part of conversion specification, so the user must enter data of the form 6/9.

8. A fatal error will be generated if the format string is ended with a white space character.
a) true
b) false

ANSWER:- a
Explanation: An error will be generated if the format string %s is ended with white space character.

9. Explain the format string "%5d%s %c"
a) five characters as a decimal integer, then reads the remaining as a string and then scans the first non-whitespace character
b) compile error
c) run-time error
d) read first five characters as a decimal and ignore the rest

ANSWER:- a
Explanation: The above format string reads the first five characters as a decimal integer, then reads the remaining as a string until a space, newline or tab is found, then reads the first non-whitespace character.

10. _____ is an optional argument that gives the maximum number of characters to be read.
a) modifiers
b) width
c) precision
d) length

ANSWER:- b
Explanation: Width is the argument that gives the maximum number of characters to be read. Few characters will be read if the scanf function encounters white space and it will stop processing further.

1. Select the correct value of i from given options
i=scanf("%d %d", &a, &b);
a) 1
b) 2
c) 3
d) No value assigned

ANSWER:- b
Explanation: i stores the number of read data from the stream. It is useful for detecting an error in data input.

2. If the user enters 1 3.2 s, what value will be returned by the scanf()?

```
scanf("%d %f %c", &s1, &s2, &s3);
```

a) 1
b) 2
c) 3
d) No return value

ANSWER:- c
Explanation: When the scanf() function completes reading all the data values, it returns number of values that are successfully read.

3. If the user enters 1 s 3.2, what value will be returned by the scanf()?

```
scanf("%d %f %c", &a, &b, &c);
```

a) 1
b) 2
c) 3
d) no return value

ANSWER:- a
Explanation: scanf() returns the number of values that are successfully read. In the above statement, only integer value is read successfully.

4. What error will be generated on using incorrect specifier for the datatype being read?
a) compile error
b) run-time error
c) logical error
d) no error

ANSWER:- b
Explanation: Using an incorrect specifier for the datatype being read will generate a run-time error.

5. What is the prototype of scanf function?
a) scanf("controlstring",arg1,arg2,arg3,....,argn);
b) scanf("control string", variable list);
c) scanf(" varible list,", control string);
d) scanf("arg1,arg2,arg3,....,argn", control string);

ANSWER:- a
Explanation: The syntax of the scanf() can be given as,scanf("control string", arg1,arg2,arg3,....,argn);

6. Control string specifies the type and format of the data that has to be obtained from the keyboard.
a) true
b) false

ANSWER:- a
Explanation: The control string specifies the type and format of the data that has to be obtained from the keyboard and store in the memory locations pointed by the arguments arg1,arg2....argn.

7. What is the qualifying input for the type specifier G?
a) floating point numbers
b) floating point numbers in exponential format
c) floating point numbers in the shorter of exponential format
d) not a type specifier

ANSWER:- c
Explanation: G is a type specifier used to take an input of floating point numbers in the shorter of exponential format.

8. scanf() is a predefined function in_____header file.
a) stdlib. h
b) ctype. h
c) stdio. h
d) stdarg. h

ANSWER:- c
Explanation: scanf() is a predefined function in "stdio.h" header file.printf and scanf() carry out input and output functions in C. These functions statements are present in the header file stdio.h.

9. What does the C statement given below says?

```
scanf("%7s",ch);
```

a) read string with minimum 7 characters.
b) read string with maximum 7 characters
c) read string exactly to 7 characters
d) read string with any number of characters

ANSWER:- b
Explanation: In the above statement the control string specifies the size of string to be 7(i.e only 7 characters can be entered in a string).

10. What is the meaning of the following C statement?

```
scanf("%[^\n]s", ch);
```

a) read all character except new line
b) read all characters
c) read only new line character
d) syntax error

ANSWER:- a
Explanation: The symbol ^ when used before a escape sequence, does not read from the console.

1. Which one of the following is correct syntax for opening a file.
a) FILE *fopen(const *filename, const char *mode)
b) FILE *fopen(const *filename)

c) FILE *open(const *filename, const char *mode)
d) FILE open(const*filename)

ANSWER:- a
Explanation: fopen() opens the named file, and returns a stream, or NULL of the attempt fails.

2. What is the function of the mode ' w+'?
a) create text file for writing, discard previous contents if any
b) create text file for update, discard previous contents if any
c) create text file for writing, do not discard previous contents if any
d) create text file for update, do not discard previous contents if any

ANSWER:- b
Explanation: w+ is a mode used to open a text file for update (i. e., writing and reading), discard previous contents if any.

3. If the mode includes b after the initial letter, what does it indicates?
a) text file
b) big text file
c) binary file
d) blueprint text

ANSWER:- c
Explanation: If the mode consists of letter b after the first letter as in, "rb" or "w+b", it indicates binary file.

4. fflush(NULL) flushes all _____
a) input streams
b) output streams
c) previous contents
d) appended text

ANSWER:- b
Explanation: fflush(FILE *stream) – fflush() causes any buffered but unwritten to be written on an Output stream. On an input stream, the effect is undefined. fflush(NULL) flushes all output streams.

5. _____removes the named file, so that a subsequent attempt to open it will fail.
a) remove(const *filename)
b) remove(filename)
c) remove()
d) fclose(filename)

ANSWER:- a
Explanation: remove(const *filename) removes the named file, so that a subsequent attempt to open it will fail. It returns non-zero of the attempt fails.

6. What is the function of FILE *tmpfile(void)?
a) creates a temporary file of mode "wb+"
b) creates a temporary file of mode "wb"
c) creates a temporary file of mode " w"
d) creates a temporary file of mode "w+"

ANSWER:- a
Explanation: A temporary file is created by tmpfile() function of mode "wb+" that will be automatically removed when closed or when the program terminates normally.

7. What does tmpfile() returns when it could not create the file?
a) stream and NULL
b) only stream
c) only NULL
d) does not return anything

ANSWER:- a
Explanation: tmpfile() returns a stream or NULL if it could not create the file.

8. Choose the right statement for fscanf() and scanf()
a) fscanf() can read from standard input whereas scanf() specifies a stream from which to read
b) fscanf() can specifies a stream from which to read whereas scanf() can read only from standard input
c) fscanf() and scanf() has no difference in their functions
d) fscanf() and scanf() can read from specified stream

ANSWER:- b
Explanation: The fscanf() is similar to the scanf() function, except that the first argument of fscanf() specifies a stream from which to read whereas scanf() can read from standard input.

9. EOF is an integer type defined in stdio. hand has a value _____
a) 1
b) 0
c) NULL
d) – 1

ANSWER:- d
Explanation: EOF is an integer type defined in stdio. hand has a value – 1.

10. fwrite() can be used only with files that are opened in binary mode.

a) true
b) false

ANSWER:- a
Explanation: fwrite() can be used to write characters, integers, or structures to a file. However, fwrite() can be used only with files opened in binary mode.

1. what is the function of fputs()?
a) read a line from a file
b) read a character from a file
c) write a character to a file
d) write a line to a file

ANSWER:- d
Explanation: The fputs() is used to write a line to a file. fputs() syntax can be written as
int fputs(const char *str, FILE *stream);

2. What does the following C code snippet mean?

```
char *gets(char *s)
```

a) reads the next input line into the array s
b) writes the line into the array s
c) reads the next input character into the array s
d) write a character into the array

ANSWER:- a
Explanation: gets() reads the next input line into the array s, terminating newline is replaced with '\0'.It returns s, or NULL if end of file or error occurs.

3. Which function will return the current file position for stream?
a) fgetpos()
b) fseek()
c) ftell()
d) fsetpos()

ANSWER:- c
Explanation: The current file position is returned by ftell() function for stream, or -1L on error.

4. Select the right explanation for the following C code snippet.

```
int fgetpos(FILE *stream, fpos_t *s)
```

a) records the current position in stream in *s
b) sets the file position for stream in *s
c) positions stream at the position recorded in *s
d) reads from stream into the array ptr

ANSWER:- a
Explanation:fgetpos() records the current position in stream in *s, for subsequent use by fsetpos() . The type fpost_t is suitable for recording such values.

5. Which functions is declared in <errno. h>?
a) fseek()
b) ftell()
c) ferror()
d) fsetpos()

ANSWER:- c
Explanation: ferror() is declared under <errno. h>. ferror() returns non-zero if the error indicator for stream is set.

6. setvbuf() and setbuf() function controls buffering for the stream.
a) true
b) false

ANSWER:- a
Explanation: setvbuf() and setbuf() controls buffering for the stream. If buff is NULL, buffering is turned off for the stream.

7. The functions vprintf(), vfprintf(), and vsprintf() are not equivalent to the corresponding printf() functions except the variable argument list.
a) true
b) false

ANSWER:- b
Explanation: The functions vprintf() , vfprintf() , and vsprintf() are similar to the corresponding printf() functions except that the variable argument list is replaced by arg.

8. The_____function reads atmost one less than the number of characters specified by size from the given stream and it is stored in the string str.
a) fget()
b) fgets()
c) fput()
d) fputs()

ANSWER:- b
Explanation: The fgets() function reads one less than the number of characters indicated by the size from the given stream and it is stored in the string str. The fgets() terminates as soon as it encounters either a newline character, EOF, or other error.

9. What does the following C code snippet mean?

```
int ungetc(int c, FILE *stream)
```

a) pushes c back onto a stream
b) deletes c from the stream
c) reads frequency of c in stream
d) no action is taken by the command

ANSWER:- a
Explanation: ungetc() pushes c back onto stream, where it will be returned on the next read. Only one character of pushback per stream is Guaranteed.

10. Choose the correct difference between getc() and fgetc().
a) If it is not a macro, it may evaluate stream more than once
b) if it is amacro, it may not evaluate stream more than once
c) if it is a macro, it may evaluate stream more than once
d) no difference between fgetc() and getc()

ANSWER:- c
Explanation: getc() is equivalent to fgetc() except that if it is a macro, it may evaluate more than once.

1. How many digits are present after the decimal in float value?
a) 1
b) 3
c) 6
d) 16

ANSWER:- c
Explanation: None.

2. Which among the following is never possible as an output for a float?
a) 3.666666
b) 3.666
c) 3
d) None of the mentioned

ANSWER:- d
Explanation: None.

3. In a 32-bit compiler, which 2 types have the same size?
a) char and short
b) short and int
c) int and float
d) float and double

ANSWER:- c
Explanation: None.

4. What is the size of float in a 32-bit compiler?
a) 1
b) 2
c) 4
d) 8

ANSWER:- c
Explanation: None.

5. Loss in precision occurs for typecasting from_____
a) char to short
b) float to double
c) long to float
d) float to int

ANSWER:- d
Explanation: None.

6. In the following C code, the union size is decided by?

```
1.    union temp
2.    {
3.        char a;
4.        int b;
5.        float c;
6.    };
```

a) char
b) int
c) float
d) both int and float

ANSWER:- d
Explanation: None.

7. %f access specifier is used for _____
a) Strings
b) Integral types
c) Floating type
d) All of the mentioned

ANSWER:- c
Explanation: None.

8. Select the odd one out with respect to type?
a) char
b) int
c) long
d) float

ANSWER:- d
Explanation: None.

1. What will be the output of the following C code?

```
#include <stdio.h>
printf("%.0f", 2.89);
```

a) 2.890000
b) 2.89
c) 2
d) 3

ANSWER:- d
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    float a = 2.455555555555;
    printf("%f", a);
}
```

a) 2.455555
b) 2.455556
c) 2.456
d) 2.46

ANSWER:- a
Explanation: None.

3. Which of the following % operation is invalid?
a) 2 % 4;
b) 2 % 4l;
c) 2 % 4f;
d) Both 2 % 4l; and 2 % 4f;

ANSWER:- c
Explanation: None.

4. Which data type is suitable for storing a number like?

10.0000000001

a) int
b) float
c) double
d) both float and double

ANSWER:- c
Explanation: None.

5. Modulus for float could be achieved by?
a) a % b
b) modulus(a, b);
c) fmod(a, b);
d) mod(a, b);

ANSWER:- c
Explanation: None.

6. Predict the data type of the following mathematical operation?

2 * 9 + 3 / 2 . 0

a) int
b) long
c) float
d) double

ANSWER:- d
Explanation: None.

7. %lf is used to display?
a) float
b) long float
c) double
d) all of the mentioned

ANSWER:- c
Explanation: None.

1. What is the sizeof(char) in a 32-bit C compiler?
a) 1 bit
b) 2 bits
c) 1 Byte
d) 2 Bytes

ANSWER:- c
Explanation: None.

2. What will be the output of the following C code?

```
#include <stdio.h>
    printf("%d", sizeof('a'));
```

a) 1
b) 2
c) 4
d) None of the mentioned

ANSWER:- c
Explanation: None.

3. Size of an array can be evaluated by _____

(Assuming array declaration int a[10];)

a) sizeof(a);
b) sizeof(*a);
c) sizeof(a[10]);

d) 10 * sizeof(a);

ANSWER:- a
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
union temp
{
    char a;
    char b;
    int c;
}t;
int main()
{
     printf("%d", sizeof(t));
     return 0;
 }
```

a) 1
b) 2
c) 4
d) 6

ANSWER:- c
Explanation: None.

5. Which of the following is not an operator in C?
a) ,
b) sizeof()
c) ~
d) None of the mentioned

ANSWER:- d
Explanation: None.

6. Which among the following has the highest precedence?
a) &
b) <<
c) sizeof()
d) &&

ANSWER:- c
Explanation: None.

7. What is the sizeof(void) in a 32-bit C?
a) 0
b) 1
c) 2
d) 4

ANSWER:- b
Explanation: None.

8. What type of value does sizeof return?
a) char
b) short
c) unsigned int
d) long

ANSWER:- c
Explanation: None.

1. Which among the following is never possible in C when members are different in a structure and union?

```
//Let P be a structure

//Let Q be a union
```

a) sizeof(P) is greater than sizeof(Q)
b) sizeof(P) is less than sizeof(Q)
c) sizeof(P) is equal to sizeof(Q)
d) none of the mentioned

ANSWER:- d
Explanation: None.

2. Which among the following is never possible in C when members in a structure are the same as that in a union?

```
//Let P be a structure

//Let Q be a union
```

a) sizeof(P) is greater than sizeof(Q)
b) sizeof(P) is equal to sizeof(Q)
c) sizeof(P) is less than to sizeof(Q)
d) none of the mentioned

ANSWER:- c
Explanation: None.

3. What will be the size of the following C structure?

```
#include <stdio.h>
struct temp
{
    int a[10];
    char p;
};
```

a) 5
b) 11
c) 41
d) 44

ANSWER:- d
Explanation: None.

4. What will be the output of the following C code?

```
#include <stdio.h>
    main()
    {
        int a = 1;
        printf("size of a is %d, ", sizeof(++a));
        printf("value of a is %d", a);
    };
```

a) size of a is 4, value of a is 1
b) size of a is 4, value of a is 2
c) size of a is 2, value of a is 2
d) size of a is 2, value of a is 2

ANSWER:- a
Explanation: None.

5. Which among the following statement is right?
a) sizeof(struct stemp*) > sizeof(union utemp*) > sizeof(char *)
b) sizeof(struct stemp*) < sizeof(union utemp*) < sizeof(char *)
c) sizeof(struct stemp*) = sizeof(union utemp*) = sizeof(char *)
d) the order Depends on the compiler

ANSWER:- c
Explanation: None.

6. What will be the output of the following C code?

```
    #include <stdio.h>
    printf("%d", sizeof(strlen("HELLOWORLD")));
```

a) Output, 4
b) Output, 10
c) Output, 16
d) Error, sizeof cannot evaluate size of a function

ANSWER:- a
Explanation: None.

7. Which of the following cannot be used inside sizeof?
a) pointers
b) functions
c) macro definition
d) none of the mentioned

ANSWER:- d
Explanation: None.

8. What will be the output of the following C code?

```
    #include <stdio.h>
    (sizeof double = 8, float = 4, void = 1)
    #define PI 3.14
    int main()
    {
        printf("%d", sizeof(PI));
    }
```

a) Output is 8
b) Output is 4
c) Output is 1
d) Error, we can't use sizeof on macro-definitions

ANSWER:- a
Explanation: None.

1. A user defined data type, which is used to assign names to integral constants is called _____
a) Union
b) Array
c) Structure
d) Enum

ANSWER:- d
Explanation: Enumeration (enum) is a user defined data type in C. It is used to assign names to integral constants. The names make a program easy to read and maintain.

2. What will be the output of the following C code?

```
#include<stdio.h>
enum colour
{
    blue, red, yellow
};
main()
{
    enum colour c;
```

```
    c=yellow;
    printf("%d",c);
}
```

a) 1
b) 2
c) 0
d) Error

ANSWER:- b
Explanation: Enum variables are automatically assigned values if no value is specified. The compiler by default assigns values starting from 0. Therefore, in the above code, blue gets 0, red gets 1 and yellow gets 2.

3. Point out the error (if any) in the following C code?

```
#include<stdio.h>
enum hello
{
    a,b,c;
};
main()
{
    enum hello m;
    printf("%d",m);
}
```

a) No error
b) Error in the statement: a,b,c;
c) Error in the statement: enum hello m;
d) Error in the statement: printf(“%d”,m);

ANSWER:- b
Explanation: In the above code, there is a semi colon given at the end of the list of variables. This results in an error. Semi colon is to be put only after the closing brace of the enum, not after the list of variables.

4. String handling functions such as strcmp(), strcpy() etc can be used with enumerated types.
a) True
b) False

ANSWER:- b
Explanation: Enumerated types are not strings. Hence it is not possible to use string handling functions with enumerated data types.

5. What will be the output of the following C code?

```
#include<stdio.h>
enum hello
{
    a,b=99,c,d=-1
};
main()
{
    enum hello m;
    printf("%d\n%d\n%d\n%d\n",a,b,c,d);
}
```

a)

1

99

100

-1

b) Error
c)

0

99

100

-1

d)

0

1

2

3

ANSWER:- c
Explanation: We can assign values to some of the symbol names in any order. All unassigned names get the value as the value of previous name plus one.

6. Pick the incorrect statement with respect to enums.
a) Two enum symbols cannot have the same value
b) Only integer constants are allowed in enums
c) It is not possible to change the value of enum symbols
d) Enum variables are automatically assigned values if no value is specified

ANSWER:- a
Explanation: The statement that two enum symbols cannot have the same value is incorrect. Any number of enum symbols can have the same value.

7. What will be the output of the following C code?

```
#include<stdio.h>
enum sanfoundry
{
    a=2,b=3.56
};
enum sanfoundry s;
main()
{
    printf("%d%d",a,b);
}
```

a) 2 3
b) 0 1
c) 2 3.56
d) Error

ANSWER:- d
Explanation: The above code will result in an error because 3.56 is not an integer constant. Only integer constants are allowed in enums.

8. What will be the output of the following C code?

```
#include<stdio.h>
enum class
{
    a,b,c
};
enum class m;
main()
{
    printf("%d",sizeof(m));
}
```

a) 3
b) Same as the size of an integer
c) 3 times the size of an integer
d) Error

ANSWER:- b
Explanation: The output will be the same as the size of an integer, that is 4 on a 32 bit platform.

9. What will be the output of the following C code?

```
#include<stdio.h>
enum hi{a,b,c};
enum hello{c,d,e};
main()
{
    enum hi h;
    h=b;
    printf("%d",h);
    return 0;
}
```

a) 2
b) 1
c) Error
d) 0

ANSWER:- c
Explanation: The code shown above results in an error: re-declaration of enumerator ‘c’. All enumerator constants should be unique in their scope.

10. What will be the output of the following C code?

```
#include<stdio.h>
enum sanfoundry
{
    a,b,c=5
};
enum sanfoundry s;
main()
{
    c++;
    printf("%d",c);
}
```

a) Error
b) 5

c) 6
d) 2

ANSWER:- a
Explanation: The above code results in an error because it is not possible to modify the value of enum constants. In the above code, we have tried to increment the value of c. This results in an error.

1. What will be the output of the following C code?

```
main()
{
    enum resut {pass, fail};
    enum result s1,s2;
    s1=pass;
    s2=fail;
    printf("%d",s1);
}
```

a) error
b) pass
c) fail
d) 0

ANSWER:- a
Explanation: The code shown above results in an error, stating : storage size of s1 and s2 unknown. There is an error in the declaration of these variables in the statement: enum result s1,s2;

2. What will be the output of the following C code?

```
#include <stdio.h>
enum example {a = 1, b, c};
enum example example1 = 2;
enum example answer()
{
    return example1;
}

int main()
{
   (answer() == a)? printf("yes"):
printf("no");
   return 0;
}
```

a) yes
b) no
c) 2
d) error

ANSWER:- b
Explanation: In the code shown above, the value of example1 is returned by the function answer. The ternary statement prints yes if this value is equal to that of 'a' and no if the value is not equal to that of 'a'. Since the value of 'a' is 1 and that returned by the function is 2, therefore no is printed.

3. What will be the output of the following C code?

```
#include<stdio.h>
#define MAX 4
enum sanfoundry
{
    a,b=3,c
};
main()
{
    if(MAX!=c)
        printtf("hello");
    else
        printf("welcome");
}
```

a) error
b) hello
c) welcome
d) 2

ANSWER:- c
Explanation: The output will be welcome. The value of the macro MAX is 4. Since 4 is equal to the value of c, welcome is printed.

4. Arithmetic operations such as addition, subtraction, multiplication and division are allowed on enumerated constants.
a) True
b) False

ANSWER:- a
Explanation: Arithmetic operations such as addition, subtraction, multiplication and division are allowed on enumerated constants. Valid arithmetic operators are +, -, *, / and %.

5. Point out the error(if any) in the following code.

```
#include<stdio.h>
enum sanfoundry
```

```
{
    a,b,c
};
enum sanfoundry g;
main()
{
    g++;
    printf("%d",g);
}
```

a) Error in the statement: a,b,c
b) Error in the statement: enum sanfoundry g;
c) Error in the statement: g++
d) No error

ANSWER:- d
Explanation: The code shown above does not result in any error. This is because, although the value of enum constants cannot be changed, yet it is possible to modify the value of the enum instance variable(that is 'g' in this case).

6. What will be the output of the following C code if input given is 2?

```
#include<stdio.h>
enum day
{
    a,b,c=5,d,e
};
main()
{
    printf("Enter the value for a");
    scanf("%d",a);
    printf("%d",a);
}
```

a) 2
b) 0
c) 3
d) Error

ANSWER:- d
Explanation: The code shown above results in an error. This is because memory is not allocated for constants in enum. Thereore ampersand(&) cannot be used with them. Also, a value has already been assigned to 'a', which cannot be changed.

7. What will be the output of the following C code if the code is executed on a 32 bit platform?

```
#include <stdio.h>
enum sanfoundry
 {
    c = 0,
    d = 10,
    h = 20,
    s = 3
} a;

int main()
{
        a = c;
         printf("Size of enum variable
= %d bytes", sizeof(a));
         return 0;
}
```

a) Error
b) Size of enum variable = 2 bytes
c) Size of enum variable = 4 bytes
d) Size of enum variables = 8 bytes

ANSWER:- c
Explanation: The size of an enum variable is equal to the size of an integer. The size of an integer on a 32 bit platform is equal to 4 bytes.

8. What will be the output of the following C code?

```
#include<stdio.h>
enum sanfoundry
{
    a=1,b,c,d,e
};
int main()
{
    printf("%d",b*c+e-d);
}
```

a) Error
b) 7
c) 2
d) 4

ANSWER:- b
Explanation: Since arithmetic operations are allowed on enum constants, hence the expression given is evaluates. b*c+e-d = 2*3+5-4 = 6+5-4 = 7

9. What will be the output of the following C code?

```
#include<stdio.h>
enum sanfoundry
{
    a,b,c=5
};
int main()
{
    enum sanfoundry s;
    b=10;
    printf("%d",b);
}
```

a) Error
b) 10
c) 1
d) 4

ANSWER:- a
Explanation: The above code results in an error. This is because it is not possible to change the values of enum constants. In the code shown above, the statement: b=10; causes the error.

10. What will be the output of the following C code?

```
#include<stdio.h>
enum sanfoundry
{
    a=1,b
};
enum sanfoundry1
{
    c,d
};
int main()
{
    enum sanfoundry1 s1=c;
    enum sanfoundry1 s=a;
    enum sanfoundry s2=d;
    printf("%d",s);
    printf("%d",s1);
    printf("%d",s2);
}
```

a) Error
b) 011
c) 110
d) 101

ANSWER:- d
Explanation: The output of the code shown above is 101. This code shows that it is possible to store the symbol of one enum in another enum variable.

1. Which of the following keywords is used to define an alternate name for an already existing data type?
a) default
b) volatile
c) typedef
d) static

ANSWER:- c
Explanation: The keyword typedef is used to define an alternate name for an already existing data type. It is mostly used for used defined data types.

2. We want to create an alias name for an identifier of the type unsigned long. The alias name is: ul. The correct way to do this using the keyword typedef is _____
a) typedef unsigned long ul;
b) unsigned long typedef ul;
c) typedef ul unsigned long;
d) ul typedef unsigned long;

ANSWER:- a
Explanation: The syntax of the keyword typedef is: keyword <existing_name> <alias_name>;
Hence if we want to create an alias name (ul) for an identifier of type unsigned long, the correct way to do this would be: typedef unsigned long ul;

3. What will be the output of the following C code?

```
#include<stdio.h>
main()
{
    typedef int a;
    a b=2, c=8, d;
    d=(b*2)/2+8;
    printf("%d",d);
}
```

a) 10
b) 16
c) 8
d) error

ANSWER:- a
Explanation: In the code shown above, the keyword typedef is used to give an alias name (a) to an identifier of the type int. The expression on

evaluation gives the answer 10. Hence the output of the code shown above is 10.

4. WWhat will be the output of the following C code? (If the name entered is: Sanfoundry)

```
#include<stdio.h>
#include<string.h>
typedef struct employee
{
    char  name[50];
    int   salary;
} e1;
void main( )
{
    printf("Enter Employee name");
    scanf("%s",e1.name);
    printf("\n%s",e1.name);
}
```

a) Sanfoundry.name
b) nSanfoundry
c) Sanfoundry
d) Error

ANSWER:- d
Explanation: The code shown above will result in an error because we have used the data type e1 (defined using the keyword typedef) in the form of an identifier.

5. The keyword typedef cannot be used to give alias names to pointers.
a) True
b) False

ANSWER:- b
Explanation: The statement given in the question is incorrect. The keyword typedef can be used to give an alias name to all data types as well as pointers.

6. What is the size of myArray in the code shown below? (Assume that 1 character occupies 1 byte)

```
typedef char x[10];
x myArray[5];
```

a) 5 bytes
b) 10 bytes
c) 40 bytes
d) 50 bytes

ANSWER:- d
Explanation: The size of myArray will be equal to 50 bytes. In the code shown above, we have defined a character array x, of size 10. Hence the output of the code shown above is 10*5 = 50 bytes.

7. We want to declare x, y and z as pointers of type int. The alias name given is: intpt The correct way to do this using the keyword typedef is:
a)

```
int typedef* intptr;
int x,y,z;
```

b)

```
typedef* intptr;
int x,y,z;
```

c)

```
int typedef* intptr;
intptr x,y,z;
```

d)

```
typedef int* intptr;
intptr x,y,z;
```

ANSWER:- d
Explanation: It shows the correct way to declare x, y and z as pointers of type int using the keyword typedef. The advantage of using typedef with pointers is that we can declare any number of pointers in a single statement.

8. Consider this statement: typedef enum good {a, b, c} hello; Which of the following statements is incorrect about hello?
a) hello is a typedef of enum good
b) hello is a structure
c) hello is a variable of type enum good
d) the statement shown above is erroneous

ANSWER:- a
Explanation: The keyword typedef is used to give

an alternate name to an existing data type. Hence hello is the new name for enum good.

9. One of the major difference between typedef and #define is that typedef interpretation is performed by the _____ whereas #define interpretation is performed by the _____
a) pre-processor, compiler
b) user, pre-processor
c) compiler, pre-processor
d) compiler, user

ANSWER:- c
Explanation: The major difference between typedef and #define is that the typedef interpretation is performed by the compiler whereas #define interpretation is performed by pre-processor.

10. What will be the output of the following C code?

```
#include<stdio.h>
int main()
{
    typedef union a
    {
        int i;
        char ch[2];
    }hello;
    hello u;
    u.ch[0] = 3;
    u.ch[1] = 2;
    printf("%d, %d", u.ch[0], u.ch[1]);
    return 0;
}
```

a) 2, 3
b) 3, 2
c) 32
d) error

ANSWER:- b
Explanation: In the code shown above, we have defined hello, which is the instance variable of a union (a) using typedef. On the execution of the code shown above, we obtain the output: 3, 2

1. There are two groups of string functions defined in the header <string.h>. What are they?
a) first group names beginning with str; second group names beginning with mem
b) first group names beginning with str; second group names beginning with is
c) first group names beginning with string; second group names beginning with mem
d) first group names beginning with str; second group names beginning with type

ANSWER:- a
Explanation: There are two groups of string functions declared under the header <string.h>. The first have names beginning with str and second have names beginning with mem.

2. What is the use of function char *strchr(ch, c)?
a) return pointer to first occurrence of ch in c or NULL if not present
b) return pointer to first occurrence of c in ch or NULL if not present
c) return pointer to first occurrence of ch in c or ignores if not present
d) return pointer to first occurrence of cin ch or ignores if not present

ANSWER:- b
Explanation: The given code char *strchr(ch, c) return pointer to first occurrence of c in ch or NULL if not present.

3. Which code from the given option return pointer to last occurrence of c in ch or NULL if not present?
a) char *strchr(ch, c)
b) char *strrchr(ch, c)
c) char *strncat(ch, c)
d) char *strcat(ch, c)

ANSWER:- b
Explanation: The function char *strrchr(ch, c) returns pointer to last occurrence of c in ch or NULL if not present.

4. Which among the given options compares atmost n characters of string ch to string s?
a) int strncmp(ch, s, n)
b) int strcmp(ch, s)
c) int strncmp(s, ch, n)
d) int strcmp(s, ch)

ANSWER:- a
Explanation: int strncmp(ch, s, n) is used to compare at most n characters of string ch to string s; return <0 if ch0 of ch >s.

5. Which among the given options is the right explanation for the statement size_t strcspn(c, s)?
a) return length of prefix of s consisting of characters not in c
b) return length of prefix of s consisting of characters present in c
c) return length of prefix of c consisting of

characters not in s
d) return length of prefix of c consisting of characters present in s

ANSWER:- c
Explanation: The function size_t strcspn(c, s) is used to return length of prefix of c consisting of characters not in s.

6. The mem functions are meant for _____
a) returning a pointer to the token
b) manipulating objects as character arrays
c) returning a pointer for implemented-defined string
d) returning a pointer to first occurrence of string in another string

ANSWER:- b
Explanation: The mem functions is used for manipulating objects as character arrays.

7. What is the function of void *memset(s, c, n)?
a) places character s into first n characters of c, return c
b) places character c into first n characters of s, return s
c) places character s into first n characters of c, return s
d) places character c into first n character of s, return c

ANSWER:- b
Explanation: The void *memset(s, c, n) places character c into first n characters of s, return s.

8. Functions whose names begin with “strn”
a) manipulates sequences of arbitrary characters
b) manipulates null-terminated sequences of characters
c) manipulates sequence of non – null characters.
d) returns a pointer to the token

ANSWER:- c
Explanation: Functions whose names begin with “strn” manipulates the sequence of non-null characters.

9. Which of the following is the right syntax to copy n characters from the object pointed to by s2 into the object pointed to by s1?
a) void *memcpy(void *s1,const void *s2,size_t n);
b) void *memcpy(void *s2, const void *s1, size_t n);
c) void memcpy(void *s1,const void *s2, size_t n);
d) void memcpy(void *s2,const void *s1,size_t n);

ANSWER:- a
Explanation: The memcpy() function copies n characters from the object pointed to by s2 into the object pointed to by s1. If copying takes place between objects that overlap, the behavior is undefined.

10. What does the following function returns void *memmove(void *s1,const void s2, size_t n);?
a) returns the value of s1
b) returns the value of s2
c) doesn’t return any value
d) returns the value of s1 and s2

ANSWER:- a
Explanation: The memmove() function copies n characters from the object pointed to by s2 into the object pointed to by s1.The memmove() function returns the value of s1.

1. Which among the following is Copying function?
a) memcpy()
b) strcopy()
c) memcopy()
d) strxcpy()

ANSWER:- a
Explanation: The memcpy() function is used to copy n characters from the object.
The code is void *memcpy(void *s1,const void *s2, size_t n).

2. Which function will you choose to join two words?
a) strcpy()
b) strcat()
c) strncon()
d) memcon()

ANSWER:- b
Explanation: The strcat() function is used for concatenating two strings, appends a copy of the string.
char *strcat(char *s1,const char *s2);

3. The _____ function appends not more than n characters.
a) strcat()
b) strcon()
c) strncat()
d) memcat()

ANSWER:- c
Explanation: The strncat() function appends not more than n characters from the array(s2) to the end of the string(s1).char *strncat(char *s1, const char *s2,size_t n);

4. What will strcmp() function do?
a) compares the first n characters of the object

b) compares the string
c) undefined function
d) copies the string

ANSWER:- b
Explanation: The strcmp() function compares the string s1 to the string s2.
int strcmp(const char *s1,const char *s2);

5. What is the prototype of strcoll() function?
a) int strcoll(const char *s1,const char *s2)
b) int strcoll(const char *s1)
c) int strcoll(const *s1,const *s2)
d) int strcoll(const *s1)

ANSWER:- a
Explanation: The prototype of strcoll() function is int strcoll(const char *s1,const char *s2).

6. What is the function of strcoll()?
a) compares the string, result is dependent on the LC_COLLATE
b) copies the string, result is dependent on the LC_COLLATE
c) compares the string, result is not dependent on the LC_COLLATE
d) copies the string, result is not dependent on the LC_COLLATE

ANSWER:- a
Explanation: The strcoll() function compares the string s1 to the string s2, both interpreted as appropriate to the LC_COLLATE category of the current locale.

7. Which of the following is the variable type defined in header string. h?
a) sizet
b) size
c) size_t
d) size-t

ANSWER:- c
Explanation: This is the unsigned integral type and is the result of the sizeof keyword.

8. NULL is the macro defined in the header string. h.
a) true
b) false

ANSWER:- a
Explanation: NULL macro is the value of a null pointer constant.

9. What will be the output of the following C code?

```
const char pla[] = "string1";
const char src[] = "string2";
printf("Before memmove place= %s, src
= %s\n", pla, src);
memmove(pla, src, 7);
printf("After memmove place = %s, src
= %s\n", pla, src);
```

a) Before memmove place= string1, src = string2
After memmove place = string2, src = string2
b) Before memmove place = string2, src = string2
After memmove place= string1, src = string2
c) Before memmove place = string2, src = string1
After memmove place= string2, src =string2
d) Before memmove place= string1, src = string2
After memmove place=string1, src = string1

ANSWER:- a
Explanation: In the C library function void *memmove(void *str1, const void *str2, size_t n) copies n characters from str2 to str1.

10. What will be the output of the following C code?

```
const char str1[]="ABCDEF1234567";
const char str2[] = "269";
len = strcspn(str1, str2);
printf("First matching character is
at %d\n", len + 1);
```

a) First matching character is at 8
b) First matching character is at 7
c) First matching character is at 9
d) First matching character is at 12

ANSWER:- a
Explanation: size_t strcspn(const char *str1, const char *str2) is used to calculate the length of the initial segment of str1, which consists entirely of characters not in str2.

1. What is the return value of strxfrm()?
a) length of the transformed string, not including the terminating null-character
b) length of the transformed string, including the terminating null-character
c) display the transformed string, not including the terminating null character
d) display the transformed string, not including the terminating null-character

ANSWER:- a
Explanation: This function returns the length of

the transformed string, not including the terminating null character.

2. Is there any function declared as strstr()?
a) true
b) false

ANSWER:- a
Explanation: This function returns a pointer to the first occurrence in s1 of any of the entire sequence of characters specified in s2, or a null pointer if the sequence is not present in s1.
char *strstr(const char *s1, const char *s2)

3. The C library function _____ breaks string s1 into a series of tokens using the delimiter s2.
a) char *strtok(char *s1,const char *s2);
b) char *strtok(char *s2,const char *s1);
c) char *strstr(char *s1,const char *s2);
d) char *strstr(char *s2,const char *s1);

ANSWER:- a
Explanation: The C library function char *strtok(char *s1, const char *s2) breaks string s1 into a series of tokens using the delimiter s2.

4. The_____function returns a pointer to the first character of a token.
a) strstr()
b) strcpy()
c) strspn()
d) strtok()

ANSWER:- d
Explanation: The strtok() function returns a pointer to the first character of a token, if there is no token then a null pointer is returned.

5. which of the following function returns a pointer to the located string or a null pointer if string is not found.
a) strtok()
b) strstr()
c) strspn()
d) strrchr()

ANSWER:- b
Explanation: The strstr() function is used to return a pointer to the located string, or if string is not found a null pointer is returned.

6. Which of the given function is used to return a pointer to the located character?
a) strrchr()
b) strxfrm()
c) memchar()
d) strchar()

ANSWER:- d
Explanation: The strchr() function is used to return a pointer to the located character if character does not occur then a null pointer is returned.

7. The strpbrk() function is used to return a pointer to the character, or a null pointer if no character from s2 occurs in s1.
a) true
b) false

ANSWER:- a
Explanation: char *strpbrk(const char *s1,const char *s2);
The first occurrence in the string s1 of any character from the string s2 is done by strpbrk().

8. What will be the output of the following C code?

```
const char str1[] = "abcdef";
const char str2[] = "fgha";
char *mat;
mat= strpbrk(str1, str2);
if(mat)
printf("First matching character: %c\n",
*mat);
else
printf("Character not found");
```

a) g
b) a
c) h
d) f

ANSWER:- d
Explanation: The strpbrk() function is used to locate the first occurrence in the string str1 of any character from the string str2.

9. What will be the output of the following C code?

```
char str1[] = "Helloworld ";
char str2[] = "Hello";
len = strspn(str1, str2);
printf("Length of initial segment
matching %d\n", len );
```

a) 6
b) 5
c) 4
d) no match

ANSWER:- b
Explanation: The length of the maximum initial segment of the string str1 which consists entirely of characters from the string str2 is computed by strspn().

10. The_____ function returns the number of characters that are present before the terminating null character.
a) strlength()
b) strlen()
c) strlent()
d) strchr()

ANSWER:- b
Explanation: The strlen() function is used to return the number of characters that are present before the terminating null character.size-t strlen(const char *s);The length of the string pointed to by s is computed by strlen().

1. What will be returned in the following C code?

```
size- t strlen(const char *s)
const char *sc;
for(sc = s; *sc!= ' \ 0 ' ; ++sc)
return(sc - s) ;
```

a) number of characters equal in sc
b) length of string s
c) doesn't return any value
d) displays string s

ANSWER:- b
Explanation: The strlen() function is used to return the length of the string s.

2. The function strsp() is the complement of strcspn().
a) true
b) false

ANSWER:- a
Explanation: Both strcspn() and strpbrk() perform the same function. Only strspn() the return values differ. The function strspn() is the complement of strcspn() .

3. What will the following C code do?

```
char * strrchr(const char *s, int c )
char ch = c;
char *sc;
for(sc = NULL; ; ++s)
if(*s == ch)
SC = 9;
i f (*s =='\O' )
return (( char *) s);
```

a) find last occurrence of c in char s[ ].
b) find first occurrence of c in char s[ ].
c) find the current location of c in char s[ ].
d) There is error in the given code

ANSWER:- a
Explanation:The strrchr() function locates the last occurrence of c (converted to a char) in the string pointed to by s. String contains null character as a terminating part of it.

4. This function offers the quickest way to determine whether two character sequences of the same known length match character for the character up to and including any null character in both.
a) strcmp()
b) memcmp()
c) strncmp()
d) no such function

ANSWER:- c
Explanation: The strncmp() function is used to compare not more than n characters (characters that follow a null character are not compared) from the array pointed to by one, to the array pointed to by other.

5. What will be the output of the following C code?

```
char str1[15];
char str2[15];
int mat;
strcpy(str1, "abcdef");
strcpy(str2, "ABCDEF");
mat= strncmp(str1, str2, 4);
if(mat< 0)
printf("str1 is not greater than str2");
else if(mat> 0)
printf("str2 is is not greater than str1");
else
printf("both are equal");
```

a) str1 is not greater than str2
b) str2 is not greater than str1
c) both are equal
d) error in given code

ANSWER:- b
Explanation: The C library function int

strncmp(const char *str1, const char *str2, size_t n) is used to compare at most the first n bytes of str1 and str2.

6. What will be the output of the following C code?

```
void *memset(void *c, int c, size-t n)
unsigned char ch = c;
unsigned char *su;
for (su = s; 0 < n; ++su, --n)
<br>
*su = ch;
<br>
```

a) store c throughout unsigned char s[n]
b) store c throughout signed char s[n]
c) find first occurrence of c in s[n]
d) find last occurrence of c in s[n]

ANSWER:- a
Explanation: This is the safe way to store the same value throughout a contiguous sequence of elements in a character array.

7. Use_____to determine the null-terminated message string that corresponds to the error code errcode.
a) strerror()
b) strstr()
c) strxfrm()
d) memset()

ANSWER:- a
Explanation: Use strerror (errcode) to determine the null-terminated message string that corresponds to the error code errcode.

8. What will be the output of the following C code?

```
const char str1[10]="Helloworld";
const char str2[10] = "world";
char *mat;
mat = strstr(str1, str2);
printf("The substring is:%s\n", mat);
```

a) The substring is:world
b) The substring is:Hello
c) The substring is:Helloworld
d) error in the code

ANSWER:- a
Explanation: The C library function char *strstr(const char *str1, const char *str2) is used to find the first occurrence of the substring str2 in the string str1.The terminating '\0' characters are not compared.

9. void *memcpy(void *dest, const void *src, size_t n) What does the following code do?
a) copies n characters from src to dest
b) copies n character from dest to src
c) transform n character from dest to src
d) transform n character from src to dest

ANSWER:- a
Explanation: In the C library function memcpy() is used to copy n characters from memory area src to memory area dest.

10. What will the given C code do?

```
int memcmp(const void *str1, const void
*str2, size_t n)
```

a) compares the first n bytes of str1 and str2
b) copies the first n bytes of str1 to str2
c) copies the first n bytes of str2 to str1
d) invalid function

ANSWER:- a
Explanation: In the C library function int memcmp(const void *str1, const void *str2, size_t n)) is used to compare the first n bytes of memory area str1 and memory area str2.

1. Which header declares several functions useful for testing and mapping characters?
a) assert.h
b) stdio.h
c) ctype.h
d) errno.h

ANSWER:- c
Explanation: The header <ctype.h> declares several functions useful for testing and mapping characters.

2. The_____function tests for any character for which isalpha or isdigit is true.
a) isxdigit()
b) isspace()
c) isalnum()
d) isupper()

ANSWER:- c
Explanation: int isalnum(int c)
The isalnum function tests for any character for which isalpha or isdigit is true.

3. What does the following C code do?

```
int iscntrl( int c);
```

a) checks if character is upper case
b) checks if character is lower case
c) tests for any control character
d) no function as such

ANSWER:- c
Explanation:
int iscntrl(int c);
The iscntrl function tests for any control character.

4. What do the following C function do?

```
int isgraph(int c);
```

a) tests for only space character
b) tests for only digit
c) tests for only lower case
d) tests for any printing character

ANSWER:- d
Explanation: int isgraph(int c);
The isgraph function tests for any printing character except space.

5. The isdigit function tests for any decimal-digit character.
a) true
b) false

ANSWER:- a
Explanation: int isdigit(int c);
This function checks whether the passed character is a decimal digit.

6. Which function returns true only for the characters defined as lowercase letters?
a) islow()
b) islower()
c) isalpa()
d) isalnum()

ANSWER:- b
Explanation: int islower(int c);
The islower function tests for any character that is a lowercase letter.

7. This function checks whether the passed character is white-space.
a) ispunct()
b) isgraph()
c) isspace()
d) isalpha()

ANSWER:- c
Explanation: The isspace function tests for any character that is a standard white-space character.

8. The standard white-space characters are the following: space (' '), form feed (' \ f '), newline (' \n') , horizontal tab (' \tr), and vertlcal tab (' \v') can be tested with function.
a) ispunct()
b) isalpha()
c) isgraph()
d) isspace()

ANSWER:- d
Explanation: The isspace function tests for any character that is a standard white-space character.

9. Which function tests for any character that is an uppercase letter.
a) iscntrl()
b) ispunct()
c) isdigit()
d) isupper()

ANSWER:- d
Explanation: isupper() returns true only for the characters defined as uppercase letters.

10. The_____function tests for any hexadecimal-digit character.
a) iscntrl()
b) ispunct()
c) isgraph()
d) isxdigit()

ANSWER:- d
Explanation: int isxdigit(int c) ;
This function tests for any hexadecimal-digit character.

1. The_____function converts an uppercase letter to the corresponding lowercase letter.
a) islower()
b) isupper()
c) toupper()
d) tolower()

ANSWER:- d
Explanation: If the argument is a character for which isupper() is true and islower() is true for another set of character, the tolower() function returns the corresponding character;otherwise, the argument is returned unchanged.

2. The toupper() function converts a _____ to the corresponding _____
a) uppercase, lowercase
b) lowercase, uppercase

c) binary, decimal
d) decimal, binary

ANSWER:- b
Explanation: The toupper() function is used to convert a lowercase letter to the corresponding uppercase letter.
If the value is a character for which islower() is true and isupper () is true for another set of characters, the toupper() function returns the corresponding character, otherwise, the value is returned unchanged.

3. fgetc, getc, getchar are all declared in _____
a) stdio. h
b) ctype. h
c) assert. h
d) stdarg. h

ANSWER:- a
Explanation: The functions getc, fgetc, getchar are all declared in stdio.h header file.

4. isalpha() function is used to detect characters both in uppercase and lowercase.
a) true
b) false

ANSWER:- a
Explanation: isalpha() function is used to test for letters in the local alphabet. For the locale, the local alphabet consists of 26 English letters, in each of two cases.

5. What will be the output of the following C code?

```
int ch= ' ';
if(isgraph(ch))
printf("ch = %c can be printed \n",ch);
else
printf("ch=%c cannot be printed \n",ch);
```

a) ch = ' ' can be printed
b) ch = ' ' cannot be printed
c) compile error
d) run-time error

ANSWER:- b
Explanation: void isgraph() function is used to check if the character has graphical representation. Graphical representations character are all those characters that can be printed except for whitespace characters, which is not considered as isgraph characters.

6. The C library function checks whether the passed character is printable.
a) isgraph()
b) isalpha()
c) isprint()
d) ispunct()

ANSWER:- c
Explanation: int isprint(int c) function is used to check whether the passed character can be printed.A control character is not a printable character.

7. What will be the output of the following C code?

```
char ch[ ] = "0xC";
if(isxdigit(ch[ ]))
printf("ch = %s is hexadecimal character
\n",ch);
else
printf("ch = %s is not hexadecimal
character \n",ch);
```

a) ch = 0xC is hexadecimal character
b) ch = 0xC is not hexadecimal character
c) compile error
d) run-time error

ANSWER:- a
Explanation: The C library function isxdigit checks whether the passed character is a hexadecimal digit.

8. Which among the following option is the full set of character class Hexadecimal digits?
a) { 0 1 2 3 4 5 6 7 8 9 A B C D E F }
b) { 0 1 2 3 4 5 6 7 8 9 a b c d e f }
c) { 0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f }
d) { 0 1 2 3 4 5 6 7 8 9}

ANSWER:- c
Explanation: digit or one of the first letters of the alphabet in either case, 'a' through 'f' and 'A' through 'F' is contained in the character class Hexadecimal digits.

9. What will be the output of the following C code?

```
int i = 0;
char c;
char str[ ] = "Little Star";
while(str[i])
{
putchar (toupper(str[i]));
i++;
}
```

a) little star
b) lITTLE sTAR
c) LITTLE STAR
d) Little Star

ANSWER:- c
Explanation: The C library function toupper converts the lowercase letter to uppercase.

10. What will be the output of the following C code?

```
int ch = '\t';
if(isprint(ch))
printf("ch = |%c| printable  \n", ch);
else
printf("ch= |%c| not printable \n",ch);
```

a) ch = |\t| printable
b) ch = |\t| not printable
c) ch = | | printable
d) ch = | | not printable

ANSWER:- d
Explanation: isprint() function is used to check whether the passed character is printable. A control character is not a printable character.

1. _____ occurs when a result is too large in magnitude to represent errors as a floating-point value of the required type.
a) underflow
b) significance loss
c) domain
d) overflow

ANSWER:- d
Explanation: An overflow occurs when a result is too large in magnitude to represent errors as a floating-point value of the required type.

2. What occurs when a result has nowhere near the number of significant digits indicated by its type.
a) domain
b) underflow
c) overflow
d) significance loss

ANSWER:- d
Explanation: A significance loss occurs when a result has nowhere near the number of significant digits indicated by its type.

3. What error occurs when a result is undefined for a given argument value?
a) significance loss
b) underflow
c) overflow
d) domain

ANSWER:- d
Explanation: A domain error occurs when a result is undefined for a given argument value.

4._____ is reported on a domain error.
a) EDOM
b) ERANGE
c) Significance loss
d) Underflow

ANSWER:- a
Explanation: EDOM is reported on a domain error.

5. ERANGE is reported on an overflow or an underflow.
a) true
b) false

ANSWER:- a
Explanation: This macro represents a range error, which occurs if an input argument is outside the range, over which the mathematical function is defined and errno is set to ERANGE.

6. What will be the output of the following C code?

```
errno = 0;
y = sqrt(2);
if(errno == EDOM)
printf(&quot;Invalid value\n&quot;);
else
printf(&quot;Valid value\n&quot;);
```

a) Invalid value
b) Valid value
c) No output
d) Compile error

ANSWER:- b
Explanation: The C library macro EDOM represents a domain error, which occurs if an input argument is outside the domain, over which the mathematical function is defined and errno is set to EDOM.

7. What will be the output of the following C code?

```
errno = 0;
y = sqrt(-10);
if(errno == EDOM)
printf(&quot;Invalid value \n&quot;);
```

```
else
printf(&quot;Valid value\n&quot;);
```

a) Invalid value
b) Valid value
c) No output
d) Compile error

ANSWER:- a
Explanation: The C library macro EDOM represents a domain error, which occurs if an input argument is outside the domain, over which the mathematical function is defined and errno is set to EDOM.

8. errno causes trouble in two subtler ways(vague and explicit).
a) true
b) false

ANSWER:- a
Explanation: errno causes trouble in two subtler ways – sometimes its specification is too vague and sometimes it is too explicit.

9. No library function will store a zero in errno.
a) true
b) false

ANSWER:- a
Explanation: Any library function can store nonzero values in errno.

10. _____ tells the compiler that this data is defined somewhere and will be connected with the linker.
a) errno
b) extern
c) variable
d) yvals

ANSWER:- b
Explanation: The C library macro extern int errno is set by system calls and some library functions in the event of an error to indicate if anything went wrong.

1. Which of the following header declares mathematical functions and macros?
a) math.h
b) assert.h
c) stdmat. h
d) stdio. h

ANSWER:- a
Explanation: The header file math.h declares all the mathematical functions and macros.

2. All the functions in this library take as a parameter and return as the output.
a) double, int
b) double, double
c) int, double
d) int, int

ANSWER:- b
Explanation: All the function in this math library takes double as a parameter and gives double as the result.

3. HUGE_VAL macro is used when the output of the function may not be _____
a) floating point numbers
b) integer number
c) short int
d) long int

ANSWER:- a
Explanation: HUGE_VAL macro is used when the result of a function may not be representable as a floating point number.

4. What error occurs if an input argument is outside the domain over which the mathematical function is defined?
a) domain error
b) range error
c) no error
d) domain and range error

ANSWER:- a
Explanation: A domain error occurs if an input argument is outside the domain over which the mathematical function is defined.

5. A range error occurs if the result of the function cannot be represented as a value.
a) int
b) short int
c) double
d) float

ANSWER:- c
Explanation: if the result of the function cannot be represented as a double value, a range error occurs. If the result overflows (the magnitude of the result is so large that it cannot be represented in an object of the specified type), the function returns the value of the macro HUGE_VAL, with the same sign as the correct value of the function.

6. If the result overflows, the function returns the value of the macro HUGE_VAL, carrying the same sign except for the _____ function as the correct value of the function.

a) sin
b) cos
c) cosec
d) tan

ANSWER:- d
Explanation: If the result overflows i.e the magnitude of the result is too large to be represented in an object of the specified type, the function returns the value of the macro HUGE_VAL, with the same sign except for the tan function as the correct value of the function.

7. If the result underflow, the function returns zero.
a) true
b) false

ANSWER:- a
Explanation: If the result underflows i.e the magnitude of the result is very small to be represented in an object of the specified type, the function returns zero.

8. For the given math function, an error occurs if the arguments are not in the range [-1, +1].

double acos(double x);

a) range error
b) domain error
c) no error
d) domain and range error

ANSWER:- b
Explanation: The acos() function is used to compute the principal value of the inverse of cosine of x. A domain error occurs for arguments not in the range [-1, +1].

9. Which function returns the arc sine in the range [-pi/2, +pi/2] radians?
a) arcsin()
b) asin()
c) sin()
d) asine()

ANSWER:- b
Explanation: The asin() function is used to compute the principal value of the inverse of sine of x. A domain error occurs for arguments not in the range [-1, +1].

10. What will be the output of the following C code?

```
double x, deg, rad;
x = 1.0;
val = 180.0 / 3.14;
deg = atan (x) * val;
printf("The arc tangent of %lf is %lf
degrees", x, deg);
```

a) The arc tangent of 1.000000 is 45.000000 degrees
b) The arc tangent of 1.000 is 45.000degrees
c) The arc tangent of 1 is 45 degrees
d) The arc tangent of 1.0000is 45.0000degrees

ANSWER:- a
Explanation: double atan(double x) returns the arc tangent of x in radians.

1. What does the given C code do?

```
double atan2 (double y, double x);
```

a) The atan2 function returns the arc tangent of x/y
b) The atan2 function returns the arc tangent of x
c) The atan2 function returns the arc tangent of y/x
d) The atan2 function returns the arc tangent of y

ANSWER:- c
Explanation: The atan2 function returns the arc tangent of y/x, in the range [-pi,+pi] radians.

2. The cos function computes the cosine of x.
a) measured in radians
b) measured in degrees
c) measured in gradian
d) measured in milliradian

ANSWER:- a
Explanation: The cos function computes the cosine of x (measured in radians).

3. The function computes the hyperbolic cosine of x.
a) cos(x)
b) cosine(x)
c) cosh(x)
d) cosineh(x)

ANSWER:- c
Explanation: The cosh() function computes the hyperbolic cosine of x. The cosh function returns the hyperbolic cosine value.

4. What error occurs if the magnitude of x is too large in sinh(double x)?
a) domain error

b) range error
c) no error
d) zero is returned

ANSWER:- b
Explanation:The sinh() function computes the hyperbolic sine of x. A range error occurs if the magnitude of x is too large.If the result overflows i.e the magnitude of the result is too large to be represented in an object of the specified type).

5. Which of the following is the correct code?
a) tanh(double x)
b) tanh double x
c) tanhdouble x
d) tanhdoublex

ANSWER:- a
Explanation: The correct code is tanh(double x). The tanh() function computes the hyperbolic tangent of x.

6. Name the function that breaks a floating-point number into a normalized fraction and an integral power of 2.
a) exp()
b) frexp()
c) ldexp()
d) modf()

ANSWER:- b
Explanation: The frexp() function breaks a floating-point number into a normalized fraction and an integral power of 2.

7. The function computes the exponential function of x.
a) exp(x)
b) frexp(x)
c) frexp x
d) exp x

ANSWER:- d
Explanation: The exp(x) function computes the exponential function of x.

8. The ldexp() function multiplies a floating-point number by an integral power of 2.
a) true
b) false

ANSWER:- a
Explanation: double ldexp(double x, int exp);
The ldexp function multiplies a floating-point number by an integral power of 2. A range error may occur. The ldexp() function returns the value of x times 2 raised to the power exp.

9. What will be the output of the following C code?

```
double log (double -x);
```

a) returns natural logarithm of x
b) range error
c) domain error
d) returns natural logarithm of -x

ANSWER:- c
Explanation: The log() function is used to compute the natural logarithm of x. If the argument is negative, a domain error occurs. A range error may occur if the argument is zero.

10. Which of the given function is a library function under the header math.h?
a) log10()
b) log20()
c) log30()
d) log50()

ANSWER:- a
Explanation: double log10 (double x);
The log10 function computes the base-ten logarithm of x. If the argument is negative, a domain error occurs. If the argument is zero, a range error may occur.

1. Which of the following statement is correct?

```
double x, y, z;
x = 5.123456;
z= modf(x, *y);
```

a) y stores integer part of x, z returns fractional part of x
b) y stores integer part of x, z returns integer part of x
c) y stores fractional part of x, z returns integer part of x
d) y stores fractional part of x, z returns fractional part of x

ANSWER:- a
Explanation: double modf(double x, double *y)
This function returns the fractional part of x, with the same sign. modf() function breaks the argument value into integer and fraction parts, each of which has the same sign as the argument. It stores the integer part as a double in the object pointed to by y.

2. A domain error occurs if x is negative and y is not an integral value for the function pow(double

x, double y).
a) true
b) false

ANSWER:- a
Explanation: The pow() function computes x raised to the power y. A domain error occurs if x is negative and y is not an integral value. A domain error occurs if the result cannot be represented when x is zero and y is less than or equal to zero. A range error may occur.

3. A function is declared as sqrt(-x) under the header file math.h, what will the function return?
a) square root of x
b) complex number
c) domain error
d) range error

ANSWER:- c
Explanation: double sqrt (double x);
The sqrt function computes the nonnegative square root of x. A domain error occurs if the argument is negative.

4. What will be the output of the following C code?

```
double x=1.2
  printf("%.1lf", ceil(x));
```

a) 1
b) 2
c) 1.0
d) 2.0

ANSWER:- d
Explanation: The ceil function returns the smallest integral value not less than x, expressed as a double.

5. What will be the output of the following C code?

```
  double x=3,y= - 6;
  printf("%lf %lf", fabs(x), fabs(y));
```

a) 3 6
b) -3 6
c) 3.0 6.0
d) 3.000000 6.000000

ANSWER:- d
Explanation: double fabs(double x);
The fabs function computes the absolute value of a floating-point number X.

6. What will be the output of the following C code?

```
double x=1.8;
printf("%.2lf",floor(x));
```

a) 2.0
b) 2.00
c) 1.0
d) 1.00

ANSWER:- d
Explanation: double floor(double x);
The floor function computes the largest integral value not greater than x.

7. double _____ (double x, double y) computes the floating-point remainder of x/y.
a) modf
b) fmod
c) ceil
d) floor

ANSWER:- b
Explanation: double fmod(doublr x, double y);
The fmod() function computes the floating-point remainder of x/y.

8. sqrt(x) function is not faster than the apparent equivalent pow(x,0.5).
a) true
b) false

ANSWER:- b
Explanation: sqrt(x) function is generally much faster than the apparent equivalent pow (x, 0.5).

9. Which of the given C function can be used instead of the apparent identity pointed to by y?

```
int x=1;
double y= 0.5 * (exp (x) + exp (-x));
```

a) cos(x)
b) cosh(x)
c) fmod(x)
d) modf(x)

ANSWER:- b
Explanation: The cosh function returns the hyperbolic cosine value. cosh(x)= 0.5 * (exp (x) + exp (-x))

10. Which function is used to recombine the fraction and exponent parts of a floating-point value after you have worked on them separately?
a) frexp()
b) exp()
c) modf()

d) ldexp()

ANSWER:- d
Explanation: ldexp() – Use this function to recombine the fraction and exponent parts of a floating-point value after you have worked on them separately.

1. _____variable type defined in the header stdlib.h is an integer type of the size of a wide character constant.
a) size_t
b) wchar_t
c) div_t
d) ldiv_t

ANSWER:- b
Explanation: wchar_t
This is an integer type of the size of a wide character constant defined under the header stdlib.h .

2. Which of the following is the correct description of EXIT_FAILURE?
a) This is the value for the exit function to return in case of failure
b) This is the value for the exit function to terminate the program
c) This is the value for the exit function to return in case of success
d) This is the value for the exit function to return in case it is the maximum value

ANSWER:- a
Explanation: The macros defined is EXIT-FAILURE which expand to integral expressions that may be used as the argument to the exit function to return unsuccessful.

3. RAND_MAX macro is the maximum value returned by the rand function.
a) true
b) false

ANSWER:- a
Explanation: RAND_MAX is the macro which expands to an integral constant expression, the value of which is the maximum value returned by the rand function.

4. Which of the given function converts the string pointed to, by the argument str to a floating-point number?
a) atof(const char *str)
b) strtod(const char *str, char **endptr)
c) atoi(const char *str)
d) atol(const char *str)

ANSWER:- a
Explanation: The atof function converts the initial portion of the string pointed to by str to double representation. Except for the behavior on the error, it is equivalent to strtod (nptr, (char **)NULL).

5. The_____function converts the initial portion of the string pointed to by, to int representation.
a) atof()
b) atoi()
c) strtod()
d) atol()

ANSWER:- b
Explanation: int atoi(const char *str); The C library function int atoi(const char *str) converts the string argument str to an integer (type int).

6. atol(const char *str) Converts the string pointed to, by the argument str.
a) to a long integer
b) to a integer
c) to a floating point number
d) to a unsigned long integer

ANSWER:- a
Explanation: long int atol(conat char *str);
The atol() function converts the initial portion of the string pointed to by str to long int representation. Except for the behavior on error, it is equivalent to strtol (nptr, (char **)NULL, 10).

7. What will be the output of the following C code?

```
char str[20];
str= "123546"; res= atof(str);
printf("String value = %s, Float value
= %f\n", str, res);
```

a) String value = 123546, Float value = 123546.0
b) String value = 123546 , Float value = 123546.000000
c) String value = 123546 , Float value = 0.000000
d) String value = 123546 , Float value = 123546.000

ANSWER:- b
Explanation: atof() function returns the converted floating point number as a double value.

8. What will be the output of the following C code?

```
char str[];
strcpy(str, "Hello");
res = atof(str);
```

```
printf("String value = %s, Float value
= %f\n", str, res);
```

a) String value = Hello, Float value = 0.000000
b) String value = Hello, Float value = 0
c) String value = “Hello” , Float value = 0.000000
d) String value = “Hello” , Float value = 0

ANSWER:- a
Explanation: atof() function returns the converted floating point number as a double value. If no valid conversion could be performed, it returns zero (0.0).

9. What will be the output of the following C code?

```
char str[20];
strcpy(str, "123456");
res = atoi(str);
printf("%s  %d\n", str, res);
```

a) 123456 0
b) 123456 0.0
c) 123456 123456
d) 123456 123456.0

ANSWER:- c
Explanation: atoi() function returns the converted integral number as an int value.

10. What will be the output of the following C code?

```
char str[] ;
strcpy(str, "Hello");
res = atoi(str);
printf(" %s  %d\n", str, res);
```

a) Hello 0.000000
b) “Hello” 0.000000
c) Hello 0
d) “Hello” 0

ANSWER:- c
Explanation: atoi() function returns the converted integral number as an int value. If no valid conversion could be performed, it returns zero.

1. What will be the output of the following C code?

```
char word[20 ] = "1.234555 WELCOME";
char *w; double dis;
dis= strtod(word, &w);
printf("The number is %lf\n", dis);
printf("String  is |%s|", w);
```

a) The number is 1.234555 String is |WELCOME|
b) The number is 1.2345550 String is
|WELCOME|
c) The number is 1.234555 String is |1.234555
WELCOME|
d) Errror

ANSWER:- a
Explanation: strtod() function is used to return the converted floating point number as a double value, else zero value (0.0) is returned.

2. Which statement is correct work reference to endptr?

```
double strtod(const char *nptr, char
**endptr);
```

a) A pointer to the starting string is stored in the object pointed to by endptr, provided that endptr is a null pointer
b) A pointer to the final string is stored in the object pointed to by endptr, provided that endptr is not a null pointer
c) A pointer to the final string is stored in the object pointed to by endptr, provided that endptr is a null pointer
d) A pointer to the starting string is stored in the object pointed to by endptr, provided that endptr is not a null pointer

ANSWER:- b
Explanation: If endptr is not NULL then endptr stores the pointer to a character after the last character.

3. Which of the following functions decomposes the input string into three pans: an initial, possibly empty, sequence of white-space characters?
a) strtod()
b) atof()
c) atol()
d) strtol()

ANSWER:- d
Explanation: The strtol() function is used to convert the initial portion of the string pointed to by, to long int representation. First, it decomposes the input string into three pans: an initial, empty, white-space characters (as specified by the isspace function).

4. The_____function is used to convert the initial portion of the string pointed to by, to unsigned long int representation.

a) strtod()
b) atol()
c) strtoul()
d) strtol()

ANSWER:- c
Explanation: unsigned long int strtoul(const char *p, char **ptr, int base) function is used to convert the initial part of the string in p to an unsigned long int value according to the given base,it must be between 2 and 36 inclusive, or be the special value 0.

5. Which of the following is the correct syntax of the function strtoul()?
a) unsigned long int strtoul(const char *n, char **ptr, int base)
b) unsigned long int strtoul(const char *n, char **ptr)
c) unsigned long int strtoul(const char *n)
d) int strtoul(const char *n)

ANSWER:- a
Explanation: usigned long int strtoul(conmt char *n, char **ptr, int base); The strtoul() function is used to convert the initial portion of the string pointed to by n to unsigned long int representation.

6. Select the right statement with reference to malloc() and calloc().
a) malloc() does not set the memory to zero whereas calloc() sets allocated memory to zero
b) malloc() sets the memory to zero whereas calloc() does not set allocated memory to zero
c) malloc() sets the memory to zero whereas calloc() sets allocated memory to zero
d) malloc() does not set the memory to zero whereas calloc() does not set allocated memory to zero

ANSWER:- a
Explanation: The difference in malloc() and calloc() is that calloc() sets the memory to zero whereas malloc()does not sets allocated memory to zero.

7. The calloc() function allocates space for an array of n objects, each of whose size is defined by size. Space is initialized to all bits zero.
a) true
b) false

ANSWER:- a
Explanation: void *calloc(size-t n, size-t size); This function is used to allocate the requested memory and returns a pointer to it.

8. Is this right explanation to the given code?

```
void *calloc(size_t n, size_t size)
#n -- This is the number of elements to be allocated.
#size -- This is the size of elements.
```

a) true
b) false

ANSWER:- a
Explanation: void *calloc(size_t n, size_t size) The calloc() function allocates space for an array of n objects, each of whose size is given by size. The space is initialized to all bits zero.

9. Which among the given function does not return a value?
a) strtoul()
b) strtol()
c) rand()
d) srand()

ANSWER:- d
Explanation: void srand(unsigned int seed);
The srand() function uses argument as a seed for a new sequence of pseudo-random numbers to be returned by subsequent calls to rand(). The srand() function returns no value.

10. Which function returns a pseudo-random integer?
a) srand()
b) rand()
c) malloc()
d) alloc()

ANSWER:- b
Explanation: int rand (void) ;
The rand() function is used to compute a sequence of pseudo-random integers in the range 0 to RAND-MAX. The rand function returns a pseudo-random integer.

1. void free(void *ptr) deallocates the memory previously allocated by a call to _____ _____ or _____
a) malloc,getenv,abort
b) calloc,malloc,exit
c) calloc,malloc,realloc
d) exit,getenv,abort

ANSWER:- c
Explanation: The free() function causes the space to be deallocated that are pointed to by ptr, that is, made available for further allocation. No action occurs if ptr is a null pointer. Otherwise, if the argument does not match a pointer earlier returned by the calloc. malloc, or realloc function.

2. The _____ function returns no value.
a) malloc()
b) realloc()
c) free()
d) calloc()

ANSWER:- c
Explanation: The calloc() function is used to return either a null pointer or a pointer to the allocated space. The malloc() function is used to return either a null pointer or a pointer to the allocated space. The realloc() function is used to return either a null pointer or a pointer to the possibly moved allocated space. The free() function returns no value.

3. What is returned by the function if the space cannot be allocated by the function malloc(), realloc() and calloc()?
a) value
b) error
c) null pointer
d) no value

ANSWER:- c
Explanation: The pointer returned points to the lowest byte address of the allocated space. A null pointer is returned, if the space cannot be allocated.

4. What is the function of the void *realloc(void *str, size-t size);?
a) allocates space for an array of str objects, each of whose size is size
b) allocates space for an object whose size is specified by size and whose value is indeterminate
c) changes the size of the object pointed to by str to the size specified by size
d) causes the space pointed to by str to be deallocated, that is, made available for further allocation

ANSWER:- c
Explanation: void *realloc(void *str, size-t size);
The realloc() function is used to change the size of the object which is pointed to by str to the size specified by size.

5. Which among the given function causes abnormal program termination ?
a) exit()
b) abort()
c) atexit()
d) getenv()

ANSWER:- b
Explanation: void abort (void);
The abort() function causes abnormal program termination to occur, and comes out directly from that place.

6. Which of the given statement is true with respect to the function atexit()?
a) The atexit() function cannot return to its caller
b) The atexit() function is used to return zero if the registration succeeds, nonzero if it fails
c) The atexit() function returns no value
d) The atexit() function causes abnormal program termination to occur

ANSWER:- b
Explanation: The prototype of function is int atexit (void (*func) (void));
The atexit() function is used to register the function pointed to by func, to be called without arguments at normal program termination.

7. The behavior is undefined if more than one call to the exit function is executed by a program.
a) true
b) false

ANSWER:- a
Explanation: The exit() function causes normal program termination to occur. The behavior is undefined if more than one call to the exit function is executed by a program.

8. Which function searches an environmenr list that are provided by the host environment?
a) getenv()
b) system()
c) srand()
d) rand()

ANSWER:- a
Explanation: The getenv() searches for the environment string pointed to, by name and returns the associated value to the string.The syntax of function is char *getenv(const char *name).

9. The system() function passes the string pointed to by string to the host environment to be executed by a command processor in an implementation-defined manner.

```
int system(const char *string);
```

a) true
b) false

ANSWER:- a
Explanation: The system function returns nonzero only if a command processor is available provided if the argument is a null pointer. The system function returns an implementation-defined value, if the argument is not a null pointer.

10. What will be the output of the following C code?

```
int main(void)
{
    int rc;
    rc = system("time");
    exit(0);
}
```

a) produces error
b) no value is returned
c) returns the time
d) nothing can be said

ANSWER:- c
Explanation: The system() function is used to pass the string pointed to by string to the host environment to be executed by a command processor in an implementation-defined manner.

1. Which of the given function is used for searching?
a) lsearch()
b) bsearch()
c) csearch()
d) qsearch()

ANSWER:- b
Explanation: bsearch() function is used to search an array of objects, the initial element is pointed by base, for an element that matches the object is pointed by key. The size of each element of the array is specified by size.

2. Which function is called repeatedly by bsearch() to compare search elements against the elements in the array?
a) mblem()
b) wctomb()
c) compar()
d) labs()

ANSWER:- c
Explanation: void *bsearch(const void *key, const void *base, size_t nitems, size_t size, int (*compar)(const void *, const void *))
The comparison function pointed to by compar() is called with two arguments that point to the key object to be searched and to an array element, in that order. The function shall return a < 0, = 0, or > 0 if the key object is considered, respectively, to be less than, to match, or to be greater than the array element.

3. The _____ function sorts an array of objects.
a) bsort()
b) hsort()
c) ssort()
d) qsort()

ANSWER:- d
Explanation: void qsort(void *base, size_t nmemb, size_t size, int (*compar)(const void *, const void*))
The qsort() function is used to sort an array of nmemb objects, the initial element of array is pointed to by base. The size of each object in array is specified by size. The contents of the array are sorted into ascending order according to a comparison function pointed to by compar.

4. Choose the correct statement.
a) bsearch() returns no value to the function
b) getenv() returns no value to the function
c) qsort() returns no value to the function
d) realloc() returns no value to the function

ANSWER:- c
Explanation: The qsort() function does not return any value.

5. Which statement is true regarding abs() and labs()?
a) The abs() function is similar to the labs() function, except that the argument and the returned value each of them have type long int
b) The abs() function is not similar to the labs() function, except that the argument and the returned value in both functions have type long int
c) The abs() function is similar to the labs() function, except that the argument and the returned value each have type short int
d) The abs() function is not similar to the labs() function, except that the argument and the returned value in both function have type short int

ANSWER:- a
Explanation: long int labs(long int i);
The abs() function is similar to the labs() function, except that the returned value and argument each are of type long int.

6. The abs() function computes the absolute value _____
a) a floating number
b) an integer number
c) a double number
d) all of the mentioned

ANSWER:- b
Explanation: int abs (int i);
The abs() function is used to compute the absolute value of an integer i. If the result cannot be represented, the behavior is undefined.

7. Which function will return the quotient and remainder on division of numerator with denominator?
a) div()
b) div_t()
c) ldiv_t()
d) labs()

ANSWER:- a
Explanation: div_t div(int num, int den);
The div () function is used to compute the quotient and remainder of the division of the numerator num by the denominator den.div_t is the structure which contains a quotient member and a remainder member.

8. What members do the structure returned by function div() contains?
a) int quot and int rem
b) float quot and float rem
c) double quot and double rem
d) no members are returned by div()

ANSWER:- a
Explanation: A structure of type div_t is returned by the function div(), contains both the quotient and the remainder. The structure contains two members,
int quot; /* quotient */
int rem; /* remainder */

9. Which of the given structure is returned by the function ldiv()?
a) div_t
b) ldiv_t
c) div_i
d) ldiv_i

ANSWER:- b
Explanation: ldiv_t is the structure returned by the function div(). ldiv() function is similar to the function div() except that the argument and the returned value is of type long int.

10. Select the multibyte character function defined under the header file stdlib.h.
a) wctomb()
b) mblen()
c) mbtowc()
d) all of the mentioned

ANSWER:- d
Explanation: wctomb(), mblen() and mbtowc() are multibyte character function defined under the header file stdlib.h. The LC_CTYPE category of the current locale affects the behaviour of these function.

1. The number of bytes contained in the multibyte character pointed to by a character is _____
a) getenv()
b) bsearch()
c) mblen()
d) qsort()

ANSWER:- c
Explanation: int mblen(const char *s, size-t n);
If s is not a null pointer, the mblen() function is used to determine the number of bytes contained in the multibyte character pointed to by s.

2. The pointer used in the mblen() function points to the _____
a) first byte of multibyte character
b) last byte of multibyte character
c) middle byte of multibyte character
d) no pointer is used in mblen function

ANSWER:- a
Explanation: The prototype of the functon mblen() is int mblen(const char *s,size_t n); The pointer ‘s’ points to the first byte of the multibyte character.

3. What is the name of the function that is used to convert multibyte character to wide character?
a) mblen()
b) mbtowc()
c) mbstowcs()
d) wcstombs()

ANSWER:- b
Explanation: mbtowc() function is used to convert the multibyte character to wide character.

4. Which function converts the wide-character string to a multibyte string?
a) wcstombs()
b) mbstowcs()
c) mbtowc()

d) mblen()

ANSWER:- a
Explanation: The C library function size_t wcstombs(char *ptr, const wchar_t *ws, size_t n) is used to convert the wide-character string was to a multibyte string starting at ptr. At most n bytes are written to ptr.

5. The C library function _____function converts the wide character to its multibyte representation.
a) mblen()
b) mbtowc()
c) wcstombs()
d) wctomb()

ANSWER:- d
Explanation: wctomb() is a C Library function which converts the wide character into its multibyte representation. It is stored at the beginning of the character array pointed to by the respective pointer.

6. The mbstowcs() function is used to return the number of array elements modified, not including a terminating zero code, if any.
a) true
b) false

ANSWER:- a
Explanation: mbstowcs() function returns the number of array elements that are modified. It does not return a terminating zero code if any. The mbstowcs() function is used to convert a sequence of multibyte characters that begins in the initial shift state from the array pointed to by, into a sequence of corresponding codes.

7. What will the given C code do?

```
#include <stdlib.h>
_Mbsave_Mbxlen={0};
int (mblen)(const char *s ,size_t n)
{
return(_Mbtowc(NULL s,n,&_Mbxlen));
}
```

a) determine length of next multibyte code
b) determine next multibyte code
c) translate multibyte string to wide char string
d) translate wide character to multibyte string

ANSWER:- a
Explanation: mblen() function is used to return the length of a multi-byte character pointed to, by the argument s. The data objects _Mbxlen and _Mbxtowc both have names with external linkage.

8. What is the purpose of the given C code?

```
#include <stdlib.h>
_Mbsave _Mbxtowc = {0};
int (mbtowc) (wchar_t *pwc, const char *a,
size_t n)
{
return (-Mbtowc (pwc, s, n, &-Mbxtowc) ) ;
}
```

a) determine length of next multibyte code
b) translates multibyte character to wide character
c) translate multibyte string to wide char string
d) translate wide character to multibyte string

ANSWER:- b
Explanation: int mbtowc(whcar_t *pwc, const char *a, size_t n)
The given function is used to convert a multibyte sequence to a wide character.

9. What is “a” in the given C code?

```
size_t wcstombs(char *s, const wchar_t *a,
size_t n)
```

a) “a” is wide character string to be converted
b) “a” is pointer to an array of char elements
c) “a” is pointer to the first byte of a multi-byte character
d) “a” C multibyte character string to be interpreted

ANSWER:- a
Explanation: The wcstombs() function is used to convert a sequence of codes that correspond to multibyte characters from the array pointed to by ‘a’ into a sequence of multibyte characters beginning in the initial shift state and stores these multibyte characters into the array pointed to by s.
size_t wcstombs(char *s, const wchar_t *a, size_t n)

10. mblen() function returns 0,if a null wide character was recognized. It returns -1 if an invalid multi-byte sequence was encountered.
a) true
b) false

ANSWER:- a
Explanation: int mblen(const char *a, size-t n);
If in the given code “a” points to a NULL character then the function returns 0 or -1 is returned if they do not form a valid multibyte character.

1. What will the code display on compiling?

```
void funccall()
{
   printf("this is funccall\n");
}
void main ()
{
    atexit(funccall);
    printf("program starts\n");
    printf("program ends\n");
}
```

a)

program starts

this is funccall

program ends

b)

this is funccall

program starts

program ends

c)

program starts

program ends

this is funccall

d) error

ANSWER:- c
Explanation: int atexit(void (*func)(void)) calls the specified function func when the program terminates. we can register termination function anywhere in the program, but it will be called at the time of the program termination.

2. What will be the output of the following C code?

```
int main ()
{
   printf("starting of program\n");
   printf("program exits\n");
   exit(0);
   printf("program ends\n");
   return(0);
}
```

a)

starting of program

program exits

program ends

b)

starting of program

program exits

c)

starting of program

program ends

d) error

ANSWER:- b
Explanation: void exit(int status) function is used to terminate the calling process immediately.

3. What will the following C code do on compilation?

```
void main ()
{
   char com[50];
   strcpy( com, "dir" );
   system(com);
}
```

a) error
b) lists down all the files and directories in the current directory under windows machine

c) terminates the calling process immediately
d) calls specified function and terminates it at the end of the program

ANSWER:- b
Explanation: int system(const char *command) function is used to pass the command name or the program name specified by command to the host environment to be executed by the command processor and returns after the command has been completed.

4. What will be the output of the following C code?

```
void main()
{
   div_t  res;
   res = div(34, 4);
   printf("quotient part = %d\n",
res.quot);
   printf("remainder part = %d\n",
res.rem);
}
```

a)

quotient part=0

remainder part=4

b)

quotient part=8

remainder part=2

c)

quotient part=4

remainder part=0

d)

quotient part=2

remainder part=8

ANSWER:- b
Explanation: div_t div(int n, int d) used to divide n (numerator) by d (denominator). div_t is a structure defined in <cstdlib>,which has two members
int quot;
in rem;

5. Initial seed is _____ for the function srand(unsigned int seed).
a) 0
b) 1
c) 00
d) 01

ANSWER:- b
Explanation: void srand(unsigned int seed); This function returns a new sequence of pseudo-random numbers using the specified seed.Initial seed is 1.

6. Which statement is true with respect to RAND_MAX?
a) specifies value for status argument to exit indicating failure
b) specifies value for status argument to exit indicating success
c) specifies maximum value returned by rand()
d) specifies maximum value returned by srand()

ANSWER:- c
Explanation: RAND_MAX specifies maximum value that has to returned by the function rand(). rand() function returns a pseudo-random number in the range 0 to RAND_MAX.

7. Which of the given function differs from the statement'errno is not neccessarily set on conversion error'?
a) atof()
b) atoi()
c) atol()
d) strtod()

ANSWER:- d
Explanation: The function atoi() and atol() are similar to strtol(), atof() is similar to strtod() except that errono is not necessarily set on conversion error.

8. void free(void *p) performs which of the following functions?
a) returns pointer to allocated space for existing contents of p
b) de-allocates space to which p points
c) to abnormally terminate the program

d) no such function defined in stdlib.h

ANSWER:- b
Explanation: void free(void *p);
This function de-allocates space to which p points(if p is not NULL).

9. Which of the given option is declared under the header file stdlib.h?
a) SEEK_CUR
b) SEEK_END
c) CLOCKS_PER_SEC
d) EXIT_SUCCESS

ANSWER:- d
Explanation: SEEK_CUR and SEEK_END is defined under stdio.h, CLOCKS_PER_SEC is defined under time.h, EXIT_SUCCESS is defined under stdlib.h. EXIT_SUCCESS specifies value for status argument to exit indicating success.

10. MB_CUR_MAX is not defined in stdlib.h.
a) true
b) false

ANSWER:- b
Explanation: MB_CUR_MAX is defined under header file stdlib.h .
MB_CUR_MAX expands into a positive integer expression.It is the maximum number of bytes in a multibyte character present in the current locale.

1. The header file assert.h of the C Standard Library defines _____ macro.
a) stderr
b) stdarg
c) setjmp
d) assert

ANSWER:- d
Explanation: The header file assert.h of the C Standard Library defines a macro called assert is used to verify assumptions made by the program and print a diagnostic message if this assumption is false.

2. What is the name of the macro that is referred by assert macro defined in assert .h?
a) STDARG
b) SETJMP
c) NDEBUG
d) STDERR

ANSWER:- c
Explanation: The header file <assert. h> defines the assert macro and refers to another macro, NDEBUG which is not defined by <assert.h>.

3. If NDEBUG is defined as a macro name, at the point where <assert.h> is included, then assert macro is defined as #define assert(ignore) ((void)0).
a) true
b) false

ANSWER:- a
Explanation: If NDEBUG is defined as a macro name at the point in the source file where is included, then the assert macro is defined as #define assert(ignore) ((void)0) .

4. The assert shall be implemented as a _____ not as an actual _____
a) function, macro
b) macro, function
c) header, macro
d) macro, header

ANSWER:- b
Explanation: assert shall be implemented as a macro and not a function, which is used to add diagnostics information in your C program. The behavior is undefined if the macro definition is suppressed to access an actual function.

5. The assert macro returns_____value.
a) integer
b) float
c) double
d) no

ANSWER:- d
Explanation: No value is returned by the macro assert.

6. The macro void assert(int expression) allows the diagnostic information to be written in which of the following files?
a) standard error file
b) output file
c) string file
d) character file

ANSWER:- a
Explanation: The macro assert prints diagnostic information on the standard error file(stderr). The assert macro writes the information about the particular call that failed (including the text of the argument, name of the source file and the source line number) on the standard error file in an implementation-defined format.

7. Which is the correct declaration of macro assert?
a) void assert(int expression);
b) void assert(float expression);
c) void assert(double expression);
d) void assert(expression);

ANSWER:- a
Explanation: The right declaration of the macro assert defined under the header file assert.h is void assert(int expression), expression can be any variable.

8. If the expression in void assert(int expression) is zero then a message is printed on stderr(standard error file).
a) true
b) false

ANSWER:- a
Explanation: When the expression in the macro void assert(int expression) is zero then the macro displays error message on stderr.

9. void assert(int expression) when the expression is evaluated to true?
a) assert returns integer value
b) assert displays error message
c) assert returns nothing
d) assert displays pattern

ANSWER:- c
Explanation: void assert(int expression), expression can be a variable or any C expression. If the expression evaluates to TRUE, assert() does nothing.

10. Which function is called by macro assert to terminate the execution?
a) exit()
b) atexit()
c) abort()
d) no function called

ANSWER:- c
Explanation: assert macro calls abort() to abnormally terminate the execution. When the expression of the macro is zero then an error message is displayed on the standard error file and then the program execution is aborted.

1. The following message is displayed in stderr.
Assertion failed: expression, file filenum, line nmn
a) true
b) false

ANSWER:- a
Explanation: The message written might be of the form
Assertion failed: expression, file filename, line nmn
This message is displayed on stderr only when the expression in the statement void assert(int expression) is zero.nmn is the line number.

2. The source filename and line number come from the preprocessor macros _____ and _____
a) _ _FILE_ _ and _ _LINE_ _
b) _ _NAME_ _ and _ _NUMBER_ _
c) _ _FILENAME_ _ and _ _NMN_ _
d) _ _FILE_ _ and _ _NUM_ _

ANSWER:- a
Explanation: The source filename and line number come from the preprocessor macros _ _FILE_ _ and _ _LINE_ _. Filename and line number are displayed in the error message when the expression of the assert is zero. It is displayed as
Assertion failed: expression, file filename, line nmn.

3. The function abort() is defined in which of the following header file?
a) stdio.h
b) stdarg.h
c) stdlib.h
d) assert.h

ANSWER:- c
Explanation: abort() function is declared inside the header file stdlib.h .This function is used to terminate the program abnormally.

4. Correct code to turn assertions ON at various places throughout a source file is _____
a)

```
#undef NDEBUG

#include
```


b)

```
#define NDEBUG

#include
```

c)

#define NDEBUG

#include

d)

#undef NDEBUG

#include

ANSWER:- a
Explanation: you can turn assertions on and off at various places throughout a source file, so to turn assertions on, you write:
#undef NDEBUG

#include

5. Correct code to turn assertions OFF at various places throughout a source file is _____
a)

#undef NDEBUG

#include

b)

#define NDEBUG

#include

c)

#define NDEBUG

#include

d)

#undef NDEBUG

#include

ANSWER:- b
Explanation: you can turn assertions on and off at various places throughout a source file, so to turn assertions off, you write:

6. Which line from the given code is the passive form?

```
#undef assert
#ifdef NDEBUG
#define assert (test) ( (void) 0)
#else
#define assert (test)
#endif
```

a) #define assert(test)
b) #define assert(test) ((void)0)
c) #ifdef NDEBUG
d) #undef assert

ANSWER:- b
Explanation: In the given code, the line #define assert (test)((void)0) is in the pasive form.

7. What will be the output of the following C code?

```
#include <assert. h>
#include <stdio . h>
 #include <stdlib.h>
void -Assert (char *mesg)
{
   fputs (mesg, stderr);
   fputs (" -- assertion failed\n" ,
stderr);
   abort () ;
}
```

a) prints only assertion message
b) program is just aborted
c) prints assertion message and aborts
d) no action takes place

ANSWER:- c
Explanation: The given function uses assert function as defined in assert.h.The program writes assertion message on the stanard error file(stderr) and then calls abort() function to terminate the execution of the program.

8. Which macro can be used to detect and report exceptional conditions?
a) extern
b) edom
c) assert

d) lbdl_min 1e-37

ANSWER:- c
Explanation: assert macro is used to detect and report exceptional conditions. It is used to write diagnostics information on standard error file.

9. What will be the output of the following C code?

```
#include <assert.h>
#include <stdio.h>
void main()
{
   int n=11;
   char str[50]="program";
   assert(n >= 10);
   printf(" output: %d\n", n);
   assert(str != NULL);
   printf("output: %s\n", str);
}
```

a) output: 11
b) error message
c)

output: 11

output: program

d) output: program

ANSWER:- c
Explanation: The condition defined in the macro assert is true hence the statements following it are displayed.

10. What will be the output of the following C code?

```
#include <assert.h>
#include <stdio.h>
void main()
{
   int n=12;
   char str[50]="";
   assert(n >= 10);
   printf(" output: %d\n", n);
   assert(str != NULL);
   printf("output: %s\n", str);
}
```

a)

output: 12

output:

b) output: 12
c)

output: 12

assertion error

d) error

ANSWER:- c
Explanation: In the given code the first condition in the assert is true, hence the statement following it is displayed. The string is initialized null therefore the second assert condition is false and error message is written on the stderr.

1. How many macros are defined in the header file stdarg.h?
a) one
b) two
c) three
d) four

ANSWER:- c
Explanation: The header file stdarg.h has three macros defined in it.

2. The header file stdarg.h defines a variable type _____
a) va_list
b) v_list
c) size_t
d) var_list

ANSWER:- a
Explanation: The variable type declared under the header file stdarg.h is va_list. This holds information required by three macros.

3. The three macros defined by stdarg.h is _____
a) start(), arg() and end()
b) var_start(), var_arg() and var_end()
c) va_start(), va_arg() and va_end()
d) v_start(), v_arg() and v_end()

ANSWER:- c
Explanation: The header file stdarg.h has three macros defined in it. They are va_start(), va_arg()

and va_end(). va_list is a variable type that holds information needed by the three macros.

4. If access to the varying arguments is desired then the called function shall declare _____ having type va_list.
a) class
b) object
c) function
d) variable

ANSWER:- b
Explanation: An object of type va_list has to be created in order to access the varying arguments.

5. Which macro retrieves the next argument in the parameter list of the function with type type?
a) type va_arg(va_list ap, type)
b) type var_arg(va_list ap, type)
c) type v_arg(va_list ap, type)
d) type val_arg(va_list ap, type)

ANSWER:- a
Explanation: type va_arg(va_list ap,type)
This macro is used to retrieve the next argument in the parameter list of the function with type type.

6. The _____ macro shall be invoked before any access to the unnamed arguments.
a) va_arg
b) va_end
c) va_list
d) va_start

ANSWER:- d
Explanation: va_start macro shall be invoked before any access to the unnamed arguments.void va-start (va-list ap, p); The va-start macro initializes ap for subsequent use by va-arg and va-end.

7. _____ macro must be called before using _____ and _____
a) va_arg, va_end and va_start
b) va_start, va_end and va_arg
c) va_end, va_arg and va_start
d) v_arg, v_end and v_start

ANSWER:- b
Explanation: va_start macro must be called before using macros va_end and va_arg.
The macro void va_start(va_list p, last_arg) initializes p variable to be used with the va_arg and va_end macros.

8. The C library macro type _____ retrieves the next argument in the parameter list of the function with type.
a) va_end
b) va_arg
c) va_start
d) no macros

ANSWER:- b
Explanation: va_arg is the C library macro defined under stdarg.h which retrieves the next argument in the parameter list of the function.

9. What is the role of the given C function?

```
void va_end(va_list ap)
```

a) allows a function with variable arguments which used the va_start macro to return
b) retrieves the next argument in the parameter list
c) initializes ap variable to be used with the va_arg and va_start macros
d) returns the next additional argument as an expression

ANSWER:- a
Explanation: void va_end(va_list ap) defined under the header file stdarg allows a function with variable arguments which used the va_start macro to return.
The result is undefined if va_end is not called before returning from the function.

10. Which header file should be included if a function has to be defined such that it can accept variable number of arguments?
a) stdlib.h
b) stdarg.h
c) assert.h
d) setjmp.h

ANSWER:- b
Explanation: stdarg.h is the header file which should be included if a function has to be defined such that it can accept variable number of arguments.

1. Which header file is used to define data formats and currency symbols?
a) setjmp.h
b) locale.h
c) stdarg.h
d) assert.h

ANSWER:- b
Explanation: locale. h is the header file which defines the location specific settings, such as data formats and currency location.

2. Which among the given macros is defined in the header file locale.h?
a) SCHAR_MAX
b) FLT_RADIX 2
c) EDOM
d) LC_CTYPE

ANSWER:- d
Explanation: LC_CTYPE is the macro defined under header file locale.h. This macro affects all character functions.

3. Which macro sets everything defined under locale. h?
a) LC_ALL
b) LC_COLLATE
c) LC_SET
d) LC_TIME

ANSWER:- a
Explanation: LC_ALL is the macro which sets everything. It is defined under header file locale.h.

4. Select the function that reads or sets location dependent information.
a) longjmp()
b) setlocale()
c) assert()
d) toupper()

ANSWER:- b
Explanation: setlocale() function reads or sets location dependent information.
The function declaration is as follows:
char *setlocale(int category, const char loc).

5. Select the correct statement.
a) LC_MONETARY affects the monetary information
b) LC_MONETARY does not affect the monetary information
c) LC_ALL does not set everything
d) LC_CTYPE affects only one character functions

ANSWER:- a
Explanation: LC_MONETARY is the macro defined under header file locale.h which affects the monetary information provided by localeconv function.

6. Which macro is used in the setlocale() function?
a) LC_SET
b) FLT_RADIX 2
c) LC_MESSAGES
d) SHRT_MAX

ANSWER:- c
Explanation: LC_MESSAGES in the function char *setlocale(char category, const char loc) is used for system responses.

7. LC_COLLATE affects strcoll() and strxfrm() functions.
a) true
b) false

ANSWER:- a
Explanation: LC_COLLATE is the macro defined under header file locale.h. It affects strcoll() and strxfrm() functions.

8. Which macro affects the strftime() function?
a) LC_TIME
b) LC_SEC
c) LC_MIN
d) LC_SET

ANSWER:- a
Explanation: LC_TIME is the macro defined under locale. h which affects the strftime() function.

9. Select the macro that affects the information provided by localeconv function.
a) LC_ALL
b) LC_COLLATE
c) LC_NUMERIC
d) LC_CTYPE

ANSWER:- c
Explanation: LC_NUMERIC is the macro defined under the header file locale.h which affects decimal point formatting and informations provided by localeconv function.

10. What is returned by the function localeconv()?
a) current location value
b) past location value
c) pointer to the last location
d) pointer to the current location

ANSWER:- d
Explanation: The function returns pointer to the struct Iconv for the current location. The function os declared as follows:
struct lconv *localeconv(void).

1. Which of the following header file defines one function longjmp(), and one variable type jmp_buf?
a) stdarg.h
b) locale.h
c) setjmp.h
d) stdlib.h

ANSWER:- c
Explanation: setjmp.h header file defines the

macro setjmp(), one function longjmp(), and one variable type jmp_buf.

2. Which of the given options is an array type used for holding information?
a) longjmp
b) setjmp
c) jmp_buf
d) no such variable

ANSWER:- c
Explanation: jmp_buf is an array type used for holding information of macro setjmp() and function longjmp().This is a variable type defined under the header file setjmp.h.

3. Which macro saves the current environment into the variable environment for later use by the function longjmp().
a) setjmp
b) longjmp
c) jmp
d) set_jmp

ANSWER:- a
Explanation: The setjmp() macro saves its calling environment in its jmp-buf argument to be used later by the longjmp() function.

4. If setjmp() macro returns directly from the macro invocation, it_____
a) returns zero
b) returns non-zero
c) produces error
d) nothing can be said

ANSWER:- a
Explanation: The setjmp() macro returns zero if the return is from a direct invocation.

5. A non-zero value is returned, if setjmp() returns from a longjmp() function call.
a) false
b) true

ANSWER:- b
Explanation: The setjmp() macro returns a nonzero value, if the return is from a call to the longjmp() function.

6. Select the correct declaration of setjmp().
a) int setjmp(jmp_buf environment)
b) int setjmp(long_jmp environment)
c) int setjmp(jmp_buf)
d) int setjmp(long_jmp)

ANSWER:- a
Explanation: Declaration of the macro setjmp() is int setjmp(jmp_buf environment). 'environment' is a the object of type jmp_buf.

7. How many times can the macro setjmp() return?
a) one time
b) two times
c) three times
d) many times

ANSWER:- b
Explanation: setjmp() macro returns twice. First time, setjmp() returns zero on its direct invocation. Second time macro returns when longjmp is called with the information written to the environment; now, it returns the value passed to longjmp as second argument.

8. longjmp() function is the only function defined under the header file setjmp.h?
a) true
b) false

ANSWER:- a
Explanation: The only function defined under the header file setjmp.h is longjmp().LONGJMP() resets the registers to the values saved in an environment.

9. Which function restores the environment saved by the most recent invocation of the setjmp() macro in the same invocation of the program?
a) jmp_buf
b) longjmp
c) jmpbuf
d) long_jmp

ANSWER:- b
Explanation: longjmp() is the only function defined under the header file setjmp.h. What setjmp() does is save the contents of the registers so that longjmp() can restore the contents later.

10. Choose the right declaration of longjmp() function.
a) void longjmp(jmp_buf environment, int value)
b) void longjmp(setjmp environment, int value)
c) void longjmp(int value, jmp_buf environment)
d) void longjmp(int value, setjmp environment)

ANSWER:- a
Explanation: The right declaration of the function longjmp() is void longjmp(jmp_buf environment, int value). This function works in pair with setjmp() macro.

1. The header file setjmp.h is used to _____
a) set location specific information

b) control low-level calls and returns to and from functions
c) handle signals reported during a program's execution
d) manipulate strings (character arrays)

ANSWER:- b
Explanation: The header file setjmp.h is used to control low-level calls and returns to and from functions.

2. The void longjmp(jmp-buf env, int val) function causes setjmp() macro to return _____ value; if val is 0.
a) zero
b) one
c) null
d) no return

ANSWER:- b
Explanation: The longjmp() function cannot cause the setjmp() to return value zero,if the variable val is zero. It always return one when the value of val is 0.

3. Which is the true statement with respect to the function longjmp()?
a) the function where setjmp() was called has terminated, then the results are undefined
b) the function where setjmp() was called has terminated, then the results are defined
c) the function where jmp_buf was called has terminated, then the results are undefined
d) the function where jmp_buf was called has terminated, then the results are defined

ANSWER:- a
Explanation: The longjmp() function restores the environment saved by the most recent invocation of the set jmp macro in the same invocation of the program, with the corresponding jmp buf argument. The behavior is undefined, if there has been no such invocation, or if the function containing the invocation of the setjmp() macro has terminated execution in the interim.

4. Which of the given statement is not true with respect to void longjmp(jmp-buf env, int val)?
a) The variable value cannot be zero
b) env is the object containing information to restore the environment at the jmp_buf's calling point
c) This function does not return any value
d) This function restores the environment saved by the most recent call to setjmp() macro

ANSWER:- b
Explanation: In the function void longjmp(jmp-buf env, int val), env is the object containing information to restore the environment at the setjmp's calling point.

5. What is the function of the given longjump(jmp_buf buf, i)?
a) go back to place buf is pointing to and return i
b) go back to place buf is pointing to and return 0
c) uses buf to remember current position and returns 0
d) uses buf to remember current position and returns i

ANSWER:- a
Explanation: In the given function longjmp(jmp_buf buf,i), it goes back to place buf is pointing to and return i.setjmp(jmp_buf buf) uses buf to remember current position to and return 0.

6. How many times does the function longjmp() returns?
a) once
b) twice
c) thrice
d) never

ANSWER:- d
Explanation: longjmp() function defined under setjmp.h header file does not return any value.

7. A less common use of setjmp.h is to create syntax similar to _____
a) errno
b) variable arguments
c) coroutines
d) retval

ANSWER:- c
Explanation: setjmp.h is sometime is used to create syntax similar to coroutines. Coroutines are computer program components.

8. What will the following C statement do?

```
#include < setjmp.h >
int setjmp(jmp_buf env);
```

a) save the current state of the registers into env
b) resets the registers to the values saved in env
c) provides a definition for env structure
d) accept variable(env) argument lists to be written

ANSWER:- a
Explanation: The given statement is used to save the current state of the registers into env. The state of the program is dependent on the contents of its memory(code, stack, globals and heap) and the contents of its registers.

9. What are the contents of the register?
a) sp, fp only
b) sp only
c) fp, pc only
d) sp, fp, pc

ANSWER:- d
Explanation: The contents of the registers includes sp, fp and cp. sp, fp and pc stands for stack pointer, frame pointer, and program counter.

10. setjmp takes a jmp_buf type and different other type variables as input.
a) true
b) false

ANSWER:- b
Explanation: setjmp takes jmp_buf type variable as the only input.

1. Select the right statement.
a) synchronous signal occurs because of the action that your program takes
b) synchronous signal occurs because of action outside your program
c) asynchronous signal occurs because of the action that your program takes
d) division by zero is asynchronous

ANSWER:- a
Explanation: Synchronous signal occurs because of the actions that your program takes.

2. What does raise functions declared in signal.h do?
a) reports a synchronous signal
b) let's you specify handling of signals
c) reports a asynchronous signal
d) doesn't let you specify handling of signals

ANSWER:- a
Explanation: The function raise defined under the header file signal. h reports a synchronous signal.

3. What is the type declared by the header file signal.h?
a) sig_atomic_t
b) sig_signal_t
c) sig_signal_h
d) sig_stomic_h

ANSWER:- a
Explanation: The only type declared in signal.h is sig_atomic_h.
This is of int type and used as a variable in signal handling.

4. Which among the given header file is used to handle different signals reported during program execution?
a) stdarg.h
b) assert.h
c) signal.h
d) setjmp.h

ANSWER:- c
Explanation: signal.h is the header file that defines one type and two functions and many macros to handle different signals reported during the execution of the program.

5. Select the macro that abnormally terminates the program.
a) SIGILL
b) SIGTERM
c) SIGABRT
d) SIGFPE

ANSWER:- c
Explanation: SIGABRT is the macro defined under the header file signal.h which terminates the program abruptly.

6. Which of the following is the correct description of the macro SIGFPE?
a) erroneous arithmetic operation such as zero divide
b) invalid access to storage
c) termination request sent to the program
d) receipt of the interactive attention signal

ANSWER:- a
Explanation: SIGFPE is the macro defined under the header file signal.h which is an erroneous arithmetic operation such as zero divide, or operation resulting in overflow.

7. _____ gives receipt of an interactive attention signals.
a) SIGILL
b) SIGTERM
c) SIGINT
d) SIGFPE

ANSWER:- c
Explanation: SIGINT gives receipt of an interactive attention signals.

8. The sig argument specifies the signal, which may be any signal except _____ and _____
a) SIG_DFL, SIG_IGN
b) SIGKILL, SIGSTOP
c) SIG_KILL, SIG_STOP
d) SIGCHLD, SIG_IGN

ANSWER:- b
Explanation: The sigset() function is used to modify signal dispositions. The sig argument specifies the signal, it can be any signal except SIGKILL and SIGSTOP.

9. void (*signal(int sig, void (*func)(int)))(int);If the value of func is SIG_IGN then _____
a) the signal will be ignored
b) default handling for that signal will occur
c) The signal() function will fail to execute
d) the signal will be ignored

ANSWER:- a
Explanation: SIG_IGN is one of the ways in signal() function in which receipt of the signal number sig is subsequently handled. The signal will be ignored if the value of func is SIG_IGN.

10. In the c library function void (*signal(int sig, void (*func)(int)))(int), which statement is true with respect to func?
a) func is a pointer to the function
b) func is pointer to sig
c) func is a static variable
d) func is a pointer that points to all type of data

ANSWER:- a
Explanation: The declaration of the function signal() is given by
void (*signal(int sig, void (*func)(int)))(int). In this 'func' is a pointer to the function defined by the programmer or one of the following predefined functions:
SIG_DFL and SIG_IGN.

1. Some types and macros defined under the header file stddef.h may be defined under other header files too.
a) True
b) False

ANSWER:- a
Explanation: There are some types and macros which are defined under various other header files, in addition to being defined under stddef.h. For example, the type size_t is defined under stdio.h, stdlib.h, string.h etc in addition to being defined under stddef.h.

2. size_t is of _____ type.
a) signed integer
b) signed character
c) unsigned integer
d) unsigned character

ANSWER:- c
Explanation: size_t, defined under stddeh, is of the type unsigned integer. It is returned by the sizeof() operator.

3. Which of the following returns a signed integer type on finding the difference between two pointers to elements in the same array?
a) __cptrdiff__
b) cptrdiff_t
c) __ptrdiff__
d) ptrdiff_t

ANSWER:- d
Explanation: ptrdiff_t is used for pointer arithmetic and array indexing. Only pointers to elements of the same array may be subtracted from each other.

4. What will be the output of the following C code?

```
#include <stddef.h>
int main(void)
{
    int num[10];
    int *p1=&num[14], *p2=&num[19];
    ptrdiff_t a = p1-p2;
    printf("%d", a);
}
```

a) 5
b) -5
c) error
d) 10

ANSWER:- b
Explanation: The type ptrdiff_t, defined under the file stddef is used to find the difference between two pointers pointing to elements of the same array. Hence the output of the above code is -5.

5. What will be the output of the following C code?

```
#include <stddef.h>
int main(void)
{
    int num[15];
    int *p1=&num['a'], *p2=&num['A'];
```

```
    ptrdiff_t d = p1-p2;
    printf("%d", d);
}
```

a) 15
b) -32
c) 32
d) error

ANSWER:- c
Explanation: The ASCII value 'a' is 97 and that of 'A' is 65. Their difference is equal to 32. Hence the output of the code shown above is 32.

6. Which of the following is not defined under the header file stddef.h?
a) size_t
b) ptrdiff_t
c) exp_t
d) null

ANSWER:- c
Explanation: size_t, ptrdiff_t and NULL are defined under the file stddef.h. There is no type called exp_t c language.

7. Point out the error (if any) in the following C code?

```
#include <stdlib.h>
#include <stdio.h>
int main(void)
{
    int* p = NULL;
    struct S *s = NULL;
    void(*f)(int, double) = NULL;
    char *ptr = malloc(15);
    if (ptr == NULL) printf("Out of
memory");
    free(ptr);
}
```

a) Error in the statement: void(*f)(int, double) = NULL;
b) Error in the statement: char *ptr = malloc(15);
c) Error in the statement: struct S*s = NULL;
d) No error

ANSWER:- d
Explanation: The code shown above does not result in an error. However, since ptr is not equal to NULL, it does not print anything.

8. A type whose alignment requirement is at least as large as that of every data type _____
a) max_align_t
b) ptrdiff_t
c) size_t
d) null

ANSWER:- d
Explanation: The alignment of the type max_align_t is at least as great as that supported by the compiler in all contexts. An extended alignment is greater than the alignment of max_align_t type. A type having an extended alignment requirement is also called over aligned.

9. The macro offset expands to a constant of type _____
a) size_t
b) print_t
c) ptrdiff_t
d) null

ANSWER:- a
Explanation: offsetof expands to a constant of type size_t. It retuns the offset of the field member from the start to of the structure type.

10. When we use multiple alignas specifiers in the same declaration, the _____ one is used.
a) first
b) strictest
c) last
d) middle

ANSWER:- b
Explanation: alignas is one of the types used to modify the alignment requirement of the object being declared. When more than one alignas specifier is used in the same declaration, the strictest one is used.

1. Which of the following library functions returns the time in UTC (Greenwich mean time) format?
a) localtime()
b) gettime()
c) gmtime()
d) settime()

ANSWER:- c
Explanation: The library function gmtime() returns the time in UTC format. To find the current time, we use the function localtime().

2. What will be the output of the following C code? (By date we mean: date, month and year)

```
#include<stdio.h>
```

```
#include<stdlib.h>
#include<time.h>
int main()
{
    time_t ct;
    time(&ct);
    printf("%s\n",ctime(&ct));
}
```

a) only current date
b) only current date and current time
c) current date, current time and the day of the week
d) only current time

ANSWER:- c
Explanation: The code shown above will print the current date, current time and the day of the week as output.

3. What will be the output of the following C code if the current system date is 6/22/2017?

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
    time_t ct;
    time(&ct);
    struct tm *mt=localtime(&ct);
    printf("%d\n",mt-> tm_mon+2);
}
```

a) 8
b) 7
c) 5
d) 6

ANSWER:- b
Explanation: Since the given date is 22nd of June, 2017. Since June is the sixth month of the year, the output will be 5. (0-Jan, 1-Feb, 2-March, 4-April, 5-May, 6-June……..)

4. What will be the output of the following C code if the current system date is 6/22/2017?

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
typedef struct tm tm;
int main()
{
    time_t ct;
    time(&ct);
    tm *mt=localtime(&ct);
    printf("%d\n",mt-> tm_year);
}
```

a) 17
b) 2017
c) error
d) 117

ANSWER:- d
Explanation: The output of the code shown above is the number of the current year, counted from the year 1900. Hence, since the current year is 2017, the output will be: 2017-1900 = 117.

5. What will be the output of the following C code if the current system date is 6/22/2017?

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
    time_t ct;
    time(&ct);
    struct tm *mt=localtime(&ct);
    printf("%d\n",mt-> tm_date);
}
```

a) 22
b) 6
c) 22/6
d) error

ANSWER:- d
Explanation: The code shown above results in an error. This is because tm_date is not a specified function under time.h.

6. The purpose of the function ctime() is that _____

a) it returns a string representing the local time
b) it returns a void string
c) it returns a string representing the time in UTC format
d) it returns a string representing the time stored in a structure

ANSWER:- a
Explanation: The library function ctime() returns a

string representation of the local time. The function asctime() returns a string representing the time stored in a structure.

7. What will be the output of the following C code?

```
#include<time.h>
int main (void)
{
    float n = time(NULL);
    printf("%.2f\n" , n);
}
```

a) time in seconds from 1 January, 1970
b) time in minutes from 1 January, 1970
c) time in seconds from 1 January, 1980
d) time in minutes from 1 January, 1980

ANSWER:- a
Explanation: The output of the code shown above will be the time in seconds from 1 January, 1970.

8. What will be the output of the following C code if the system date is 6/2/2017(Friday)?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *local, *gm;
    time_t t;
    t=time(NULL);
    local=localtime(&t);
    printf("%d",local->tm_wday);
    return 0;
}
```

a) 6
b) 5
c) error
d) 0

ANSWER:- b
Explanation: Since the given question has clearly specified that the current weekday at the time of execution of the code is Friday, the output of the code shown above is 5. (0-Sunday, 1-Mon, 2-Tue, 3-Wed, 4 – Thurs, 5-fri, 6-Sat)

9. Which of the following functions returns a pointer to a string representing the date and time stored in a structure?
a) ctime()
b) time()
c) asctime()
d) localtime()

ANSWER:- c
Explanation: The function asctime() returns a pointer to a string representing the date and time stored in a structure.

10. What will be the output of the following C code if it is executed on 2nd January, 2017 (system date)?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *local, *gm;
    time_t t;
    t=time(NULL);
    local=localtime(&t);
    printf("%d",local->tm_yday);
    return 0;
}
```

a) 0
b) 1
c) 2
d) Error

ANSWER:- b
Explanation: The output of the code shown above is 1. In the question, it is given that the current date is 2nd January, 2017. (1st Jan – 0, 2nd Jan – 1, 3rd Jan – 2)

1. Which of the following format specifiers is used to represent the name of the time zone?
a) %A
b) %B
c) %H
d) %Z

ANSWER:- d
Explanation: The format specifier %Z is used to specify the name of the time zone. %A- full weekday, %B- full month name, %H-hours(0-23).

2. What will be the output of the following C code if the system time is 4:27 PM?

```
#include<stdio.h>
#include<time.h>
int main()
{
```

```
    struct tm *ptr;
    time_t t;
    char str[100];
    t = time(NULL);
    ptr = localtime(&t);
    strftime(str,100,"%H %p %M
minutes",ptr);
    puts(str);
    return 0;
}
```

a) 16 27 minutes
b) 4 27 minutes
c) 16 PM 27 minutes
d) 4 PM 27 minutes

ANSWER:- c
Explanation: %H is a format specifier used to represent the hours (0-23) and %I specifies the hours in the format(0-12), %p is used to specify AM or PM, %M is used to specify the minutes. Hence output will be: 16 PM 27 minutes

3. What will be the output of the following C code if the system date is 8/22/2016?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *ptr;
    time_t t;
    char str[100];
    t = time(NULL);
    ptr = localtime(&t);
    strftime(str,100,"%B",ptr);
    puts(str);
    return 0;
}
```

a) 9
b) August
c) Aug
d) Error

ANSWER:- b
Explanation: %B is a format specifier which is used to specify the full month name. Hence the output will be August.

4. What will be the output of the following C code if the system date is 6/2/2017 (Friday)?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *ptr;
    time_t t;
    char str[100];
    t = time(NULL);
    ptr = localtime(&t);
    strftime(str,100,"%A",ptr);
    puts(str);
    return 0;
}
```

a) Error
b) Fri
c) Friday
d) 6

ANSWER:- c
Explanation: %A specifies the full weekday. Hence the output is Friday.

5. Which of the following library functions is used to read location dependent information?
a) localtime()
b) localeconv()
c) localcon()
d) local()

ANSWER:- b
Explanation: localeconv() is used to read the location dependent information. The function used to find the current time is localtime().

6. Which of the following functions is used to convert the date and time into a calendar format?
a) difftime()
b) clock()
c) mktime()
d) ctime()

ANSWER:- c
Explanation: The function mktime() is used to convert the date and time into a calendar format. The function difftime() is used to find the difference between two specified timings, the function clock() is used to return the number of ticks.

7. What will be the output of the following C code?

```
#include<stdio.h>
#include<time.h>
```

```
main()
{
    struct tm t;
    time_t tc;
    t.tm_year=2017-1900;
    t.tm_mday=25;
    t.tm_mon=5;
    t.tm_hour=1;
    t.tm_min=30;
    t.tm_sec=0;
    t.tm_isdst=0;
    tc=mktime(&t);
    printf(ctime(&tc));
}
```

a) Sun Jun 25 01:30:00 2017
b) Sun June 25 1:30 2017
c) Sun Jun 25 1:30 117
d) Sun June 25 1:30:00 117

ANSWER:- a
Explanation: The function mktime() converts the time and date into a calendar format. Hence the output of the code shown above is: Sun Jun 25 1:30:00 2017.

8. The value of tm_isdst is ___ when DST(Daylight Savings Time) is in effect, _____ when DST is not in effect and _____ when the DST status is unknown.
a) -1, 1, 0
b) 1, 0, -1
c) 0, 1, -1
d) 1, -1, 0

ANSWER:- b
Explanation: The value of tm_isdst is 1 when Daylight Savings Time is in effect, 0 when DSP is not in the effect and -1 when DST status is not known.

9. The library function clock() returns the number of _____ elapsed since the start of the program.
a) minutes
b) clock ticks
c) milli-seconds
d) micro-seconds

ANSWER:- b
Explanation: The library function clock() returns the number of clock ticks elapsed since the start of the program. To get the number of seconds used by the CPU, we should divide by CLOCKS_PER_SEC.

10. What will be the output of the following C code if the name entered is “TOM” and time taken to enter this name is 2 seconds?

```
#include <stdio.h>
#include <time.h>
int main ()
{
        time_t time1,time2;
        char get_input [256];
        double dif_sec;
        time (&time1);
        printf ("Please enter the name
of your pet: ");
        gets (get_input);
        time (&time2);
        dif_sec = difftime (time2,time1);
        printf ("%.2f\n", dif_sec );
        return 0;
}
```

a) Error
b) 2
c) 2.0
d) 2.00

ANSWER:- d
Explanation: The library function difftime() returns the difference in seconds between time1 and time 2. Since the format specifier is %.2f, the output will be 2.00.

1. What will be the output of the following C code, if the system date is 6/23/2017?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *local;
    time_t t;
    t=time(NULL);
    local=localtime(&t);
    printf("%d",local->tm_mday);
    return 0;
}
```

a) 6
b) 22
c) 23
d) error

ANSWER:- c
Explanation: tm_mday returns the day of the month in terms of an integer (date). Hence the output of the code shown above will be 23 (since the system date is 6/23/2017).

2. What will be the output of the following C code, if the system date is 6/24/2017?

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
    time_t ct;
    time(&ct);
    printf("%s\n",(&ct));
}
```

a) Error
b) Junk value
c) 6
d) June

ANSWER:- b
Explanation: The output to the above code will be some junk value (not in the human readable form). This is because we have not used the ctime() to convert the date and time into a string.

3. What is the meaning of the following C code if output is 0?

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *local;
    time_t t;
    t=time(NULL);
    local=localtime(&t);
    printf("%d",local->tm_isdst);
    return 0;
}
```

a) DST is in effect
b) DST status is unknown
c) DST is not in effect
d) DST is corresponding with local time

ANSWER:- c
Explanation: Daylight Savings time is in effect when the value of tm_isdt is 1. It is not in effect when the value of tm_isdst is 0 and it's status is unknown when the value of tm_isdst is -1.

4. The following C code results in an error. State whether true or false.

```
#include<stdio.h>
#include<time.h>
int main()
{
    struct tm *local;
    time_t t;
    t=time(NULL);
    local=asctime(localtime(&t));
    printf("%d",local->tm_wday);
    return 0;
}
```

a) True
b) False

ANSWER:- b
Explanation: Although the code shown above does not give the correct answer, it does not result in an error.

5. Which of the following format specifiers is used to specify whether the given time is in AM or PM?
a) %P
b) %I
c) %X
d) %p

ANSWER:- d
Explanation: %p (small letter) is used to specify whether the given time is in AM or PM. %I is used to represent the number of hours in the 12 hour clock format. %X is used to represent the standard time zone.

6. What will be the output of the following C code if the system time is 1:52 PM (Sunday)?

```
#include<stdio.h>
#include<time.h>
int main()
{
```

```
    struct tm *ptr;
    time_t t;
    char str[100];
    t = time(NULL);
    ptr = localtime(&t);
    strftime(str,100,"%I",ptr);
    puts(str);
    return 0;
}
```

a) 13
b) 01
c) 1
d) error

ANSWER:- b
Explanation: %I is a format specifier which is used to represent the number of hours in 12 hour clock format. Hence the output of the code shown above will be 01.

7. Which of the following format specifiers is used to represent the hours in the 24 hour clock (0-23) format?
a) %I
b) %H
c) %i
d) %h

ANSWER:- b
Explanation: %H is a format specifier used to represent the number of hours in 24 hour clock format.

8. What will be the output of the following C code?

```
#include <stdio.h>
#include <time.h>
int main ()
{
          double d;
          d = difftime (5,17);
          printf ("%.2f\n", d );
          return 0;
}
```

a) 12.00
b) -12.00
c) Error
d) 12

ANSWER:- b
Explanation: The library function difftime(time1,time2) is used to find the difference between time1 and time2 such as: time1-time2. Hence the output of the code shown above will be 5-17=12.00.

9. The value returned by the library function mktime(), on failure is _____
a) -1
b) 0
c) 1
d) -2

ANSWER:- a
Explanation: The library function mktime() converts the date and time into calendar format and returns the value -1 on the failure of this action.

10. Which of the following is defined under the header file time.h?
a) strnct()
b) fabs()
c) iscntrl()
d) null

ANSWER:- d
Explanation: NULL is defined under the header file time.h, in addition to being defined under many other header files. The function strncat() is defined under string.h. The function fabs() is defined under math.h. The function iscntrl() is defined under ctye.h.

1. The maximum value of a signed char is not the same as the maximum value of an unsigned char.
a) True
b) False

ANSWER:- a
Explanation: The maximum value of a signed char is 127 and that the maximum value of an unsigned char is 255. Thus it is clear that the maximum value of a signed char is not the same as the maximum value of an unsigned char.

2. What will be the output of the following C code?

```
#include <stdio.h>
#include <limits.h>
int main()
{
    printf("The minimum value of LONG
= %lf\n", LONG_MIN);
    return 0;
}
```

a) -2147483648 (The minimum value of LONG)
b) 0.000000
c) 0
d) error

ANSWER:- b
Explanation: Since the format specifier %lf is used, 0.000000 is given as the output. If we want the output to be the minimum value of LONG, we should use the format specifier %ld is used.

3. The macro _____ defines the number of bits in a byte, which is equal to _____
a) CHAR_BIT, 4
b) CHAR_BYTE, 8
c) CHAR_BIT, 8
d) CHAR_BYTE, 4

ANSWER:- c
Explanation: The macro CHAR_BIT defines the number of bits in a byte, which is equal to 8.

4. What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    printf("%f",FLT_MIN);
}
```

a) Minimum negative value of float
b) Maximum positive value of float
c) Error
d) Minimum positive value of float

ANSWER:- c
Explanation: The macro FLT_MIN is not defined under the header file limits.h. To use any macro relate to a floating point value or a double value, we should include the header file float.h. Since this header file is not included in the code shown above, it will result in an error.

5. The macro MB_LEN_MAX is used to find _____
a) Maximum number of bytes in a multi-byte character
b) Whether the given function is valid or not
c) The maximum time taken for the execution of a particular function
d) Maximum number of bits in a multi-byte character

ANSWER:- a
Explanation: The macro MB_LEN_MAX is defined under the header file limits.h and is used to find the maximum number of bytes in a multi-byte character.

6. The value of CHAR_MAX will be equal to SCHAR_MAX when _____
a) char represents positive value
b) char represents value equal to 0
c) char represents negative value
d) char represents an exponential value

ANSWER:- c
Explanation: The macro CHAR_MAX defines the maximum value for type char and its value is equal to SCHAR_MAX is char represents negative value, otherwise UCHAR_MAX.

7. What will be the output of the following C code, given that the maximum value of signed char is 127 and that of unsigned char is 255.

```
#include<stdio.h>
#include<limits.h>
#define SCHAR_MAX
main()
{
    printf("%d",SCHAR_MAX+1);
}
```

a) 256
b) Error
c) 1
d) 128

ANSWER:- c
Explanation: The output of the code shown above will be equal to 1. This is because we have used the macro define to define a variable known as SCHAR_MAX (assigned a value of zero). Had this line not been a part of the above code, the output would be 128.

8. The macro which is used to find the maximum value of an unsigned integer is _____
a) UNINT_MAX
b) UNSINT_MAX
c) UINT_MAX
d) UNIT_MAX

ANSWER:- c
Explanation: The macro which is used to define the maximum value on an unsigned integer is: UINT_MAX. To find the maximum value on a signed integer, we use the macro: SINT_MAX.

9. To find the maximum value of an object of type unsigned long long int, we use the macro
_____
a) ULINT_MAX
b) ULLINT_MAX
c) ULONG_MAX
d) ULLONG_MAX

ANSWER:- d
Explanation: The macro ULLONG_MAX is used to find the maximum value of unsigned long long int type. The macro ULONG_MAX is used to find the maximum value of unsigned long type.

10. Which of the following macros is defined under the header limits.h?
a) FLT_ROUNDS
b) USHRT_MAX
c) DBL_MAX
d) DECIMAL_DIG

ANSWER:- b
Explanation: the macros FLT_ROUNDS, DBL_MAX and DECIMAL_DIG are defined under the header float.h. The macro USHRT_MAX is defined under the header limits.h.

1. What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    if(UCHAR_MAX<=SCHAR_MAX)
        printf("hello");
    else
        printf("good");
}
```

a) error
b) hello
c) good
d) hellogood

ANSWER:- c
Explanation: Since UCHAR_MAX (255) is greater then SCHAR_MAX (127), the output of the code shown above will be “good”.

2. Which of the following macros is not defined?
a) ULONG_MIN
b) LONG_MIN
c) ULONG_MAX
d) LONG_MAX

ANSWER:- a
Explanation: The macro ULONG_MIN is not defined whereas the macros LONG_MIN, ULONG_MAX and LONG_MAX are all defined in the header file limits.h.

3. Given that the value of SHRT_MAX is equal to 32767 and that of SHRT_MIN is equal to -32768, What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    int d;
    d=SHRT_MAX + SHRT_MIN+1;
    printf("%d",d);
}
```

a) -1
b) error
c) 1
d) 0

ANSWER:- d
Explanation: SHRT_MIN is defined as –SHRT_MAX – 1, hence the output of the code shown above will be zero.

4. What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    int d;
    d=CHAR_MIN;
    printf("%d",d);
}
```

a) 0
b) -128
c) error
d) -1

ANSWER:- b
Explanation: Since the minimum value of CHAR_MIN is equal to -128, hence the output of the code shown above is -128.

5. _____ defines the minimum value for a short integer.

a) SHINT_MIN
b) SHRT_MIN
c) SINT_MIN
d) SHORT_MIN

ANSWER:- b
Explanation: to define the minimum value for a short integer, we use the macro SHRT_MIN (defined under the header file limits.h).

6. The macro definition of INT_MIN is _____
a) –INT_MAX – 1
b) INT_MAX – 1
c) –INT_MAX + 1
d) INT_MAX + 1

ANSWER:- a
Explanation: The macro definition of INT_MIN is – INT_MAX – 1.

7. What will be the output of the following C code?

```
Maximum value of int   =  1000
Maximum value of float = 5000
Maximum value of short int = 327
Minimum value of short int = -328
#include<stdio.h>
#include<limits.h>
#include<float.h>
main()
{
    short int d;
    d=INT_MAX + FLT_MAX;
    printf("%d",d);
}
```

a) error
b) 6000
c) 327
d) -328

ANSWER:- c
Explanation: Since the data type in which we store the result of the expression INT_MAX + FLT_MAX is of the type short int, the output of the code shown above will be the maximum of short int. Hence the output is 327.

8. The macros defined under the header file limits.h are not defined under any other header file.
a) False
b) True

ANSWER:- b
Explanation: Macros such as INT_MAX, ULONG_MAX etc which are defined under the header file limits.h are not defined under any other header file.

9. What will be the output of the following C code if the value of UCHAR_MAX is 127?

```
#include<stdio.h>
#include<limits.h>
int main()
{
    int d;
    d=CHAR_MAX;
    printf("%c",d);
}
```

a) Error
b) 127
c) Alphabet corresponding to the value 127
d) Junk value

ANSWER:- d
Explanation: The output of the code shown above will be some junk value. This is because we have used the format specifier %c in the print statement. Had we used the format specifier %d, the output would have been 127 (maximum value of char).

10. The minimum value of CHAR_BIT is equal to _____
a) 2
b) 4
c) 8
d) 16

ANSWER:- c
Explanation: The minimum value of CHAR_BIT is equal to 8. This is because the macro CHAR_BIT returns the maximum number of bits in a char object. 1 byte contains 8 bits. Hence the value of CHAR_BIT cannot be less than 8.

1. What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    if(UCHAR_MAX<=SCHAR_MAX)
        printf("hello");
    else
        printf("good");
}
```

a) error
b) hello
c) good
d) hellogood

ANSWER:- c
Explanation: Since UCHAR_MAX (255) is greater then SCHAR_MAX (127), the output of the code shown above will be “good”.

2. Which of the following macros is not defined?
a) ULONG_MIN
b) LONG_MIN
c) ULONG_MAX
d) LONG_MAX

ANSWER:- a
Explanation: The macro ULONG_MIN is not defined whereas the macros LONG_MIN, ULONG_MAX and LONG_MAX are all defined in the header file limits.h.

3. Given that the value of SHRT_MAX is equal to 32767 and that of SHRT_MIN is equal to -32768, What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    int d;
    d=SHRT_MAX + SHRT_MIN+1;
    printf("%d",d);
}
```

a) -1
b) error
c) 1
d) 0

ANSWER:- d
Explanation: SHRT_MIN is defined as –SHRT_MAX – 1, hence the output of the code shown above will be zero.

4. What will be the output of the following C code?

```
#include<stdio.h>
#include<limits.h>
main()
{
    int d;
    d=CHAR_MIN;
    printf("%d",d);
}
```

a) 0
b) -128
c) error
d) -1

ANSWER:- b
Explanation: Since the minimum value of CHAR_MIN is equal to -128, hence the output of the code shown above is -128.

5. _____ defines the minimum value for a short integer.
a) SHINT_MIN
b) SHRT_MIN
c) SINT_MIN
d) SHORT_MIN

ANSWER:- b
Explanation: to define the minimum value for a short integer, we use the macro SHRT_MIN (defined under the header file limits.h).

6. The macro definition of INT_MIN is _____
a) –INT_MAX – 1
b) INT_MAX – 1
c) –INT_MAX + 1
d) INT_MAX + 1

ANSWER:- a
Explanation: The macro definition of INT_MIN is –INT_MAX – 1.

7. What will be the output of the following C code?

```
Maximum value of int   =  1000
Maximum value of float = 5000
Maximum value of short int = 327
Minimum value of short int = -328
#include<stdio.h>
#include<limits.h>
#include<float.h>
main()
{
    short int d;
    d=INT_MAX + FLT_MAX;
    printf("%d",d);
}
```

a) error
b) 6000
c) 327
d) -328

ANSWER:- c
Explanation: Since the data type in which we store the result of the expression INT_MAX + FLT_MAX is of the type short int, the output of the code shown above will be the maximum of short int. Hence the output is 327.

8. The macros defined under the header file limits.h are not defined under any other header file.

a) False
b) True

ANSWER:- b
Explanation: Macros such as INT_MAX, ULONG_MAX etc which are defined under the header file limits.h are not defined under any other header file.

9. What will be the output of the following C code if the value of UCHAR_MAX is 127?

```
#include<stdio.h>
#include<limits.h>
int main()
{
    int d;
    d=CHAR_MAX;
    printf("%c",d);
}
```

a) Error
b) 127
c) Alphabet corresponding to the value 127
d) Junk value

ANSWER:- d
Explanation: The output of the code shown above will be some junk value. This is because we have used the format specifier %c in the print statement. Had we used the format specifier %d, the output would have been 127 (maximum value of char).

10. The minimum value of CHAR_BIT is equal to _____
a) 2
b) 4
c) 8
d) 16

ANSWER:- c
Explanation: The minimum value of CHAR_BIT is equal to 8. This is because the macro CHAR_BIT returns the maximum number of bits in a char object. 1 byte contains 8 bits. Hence the value of CHAR_BIT cannot be less than 8.

1. Local variables are stored in an area called _____
a) Heap
b) Permanent storage area
c) Free memory
d) Stack

ANSWER:- d
Explanation: Local variables are stored in an area called stack. Global variables, static variables and program instructions are stored in the permanent storage area. The memory space between these two regions is known a heap.

2. The size of both stack and heap remains the same during run time.
a) True
b) False

ANSWER:- b
Explanation: Memory can be allocated or de-allocated during the run time in the heap region. Memory bindings (allocation and de-allocation) are established and destroyed during the execution of the program. Hence we can see that the size of heap does not remain same during run time. However, the size of stack remains the same.

3. Choose the statement which is incorrect with respect to dynamic memory allocation.
a) Memory is allocated in a less structured area of memory, known as heap
b) Used for unpredictable memory requirements
c) Execution of the program is faster than that of static memory allocation
d) Allocated memory can be changed during the run time of the program based on the requirement of the program

ANSWER:- c
Explanation: Execution of the program using dynamic memory allocation is slower than that using static memory allocation. This is because in dynamic memory allocation, the memory has to be allocated during run time. This slows down the execution of the program.

4. Which of the following header files must necessarily be included to use dynamic memory allocation functions?
a) stdlib.h
b) stdio.h
c) memory.h
d) dos.h

ANSWER:- a
Explanation: stdlib.h is a header file which stands for the standard library. It consists of the declaration for dynamic memory allocation functions such as malloc(), calloc(), realloc() and free.

5. The type of linked list in which the node does not contain any pointer or reference to the previous node is _____
a) Circularly singly linked list
b) Singly linked list
c) Circular doubly linked list

d) Doubly linked list

ANSWER:- b
Explanation: A singly linked list is one in which each node has two fields, namely data field and pointer field. Data field stores the data and the pointer field points to the address of the next node.

6. Which of the following is an example for non linear data type?
a) Tree
b) Array
c) Linked list
d) Queue

ANSWER:- a
Explanation: A data structure is said to be linear if its elements form a sequence or a linear list. For example array, linked list, queue, stack etc. Elements in a non linear data structure do not form a sequence. For example Trees, graphs etc.

7. Queue data structure works on the principle of _____
a) Last In First Out (LIF0)
b) First In Last Out (FILO)
c) First In First Out (FIFO)
d) Last In Last Out (LILO)

ANSWER:- c
Explanation: Queue is a linear data structure which works on the principle of first in first out. This means that the first element to be inserted in a queue will be the first one to be removed.

8. Which of the following is an example of static memory allocation?
a) Linked list
b) Stack
c) Queue
d) Array

ANSWER:- d
Explanation: Array is an example of static memory allocation whereas linked list, queue and stack are examples for dynamic memory allocation.

9. Array is preferred over linked list for the implementation of _____
a) Radix sort
b) Insertion sort
c) Binary search
d) Polynomial evaluation

ANSWER:- c
Explanation: When we try to implement binary search using linked list, the traversal steps per element increases in order to find the middle element. Thus, this process is slow and inefficient. This process is much faster using an array, hence it is preferable to use an array for the implementation of binary search.

10. The advantage of using linked lists over arrays is that _____
a) Linked list is an example of linear data structure
b) Insertion and deletion of an element can be done at any position in a linked list
c) Linked list can be used to store a collection of homogenous and heterogeneous data types
d) The size of a linked list is fixed

ANSWER:- b
Explanation: Insertion and deletion in a linked list can be done at any position. On the other hand, in an array, to insert an element at a specific position, the rest of the elements have to be moved one position to the left and to delete an element, all the elements after the deleted element have to be moved one position to the right.

1. What will be the output of the following C code if the input entered as first and second number is 5 and 6 respectively?

```
#include<stdio.h>
#include<stdlib.h>
main()
{
    int *p;
    p=(int*)calloc(3*sizeof(int));
    printf("Enter first number\n");
    scanf("%d",p);
    printf("Enter second number\n");
    scanf("%d",p+2);
    printf("%d%d",*p,*(p+2));
    free(p);
}
```

a) 56
b) Address of the locations where the two numbers are stored
c) 57
d) Error

ANSWER:- d
Explanation: The above code results in an error. This is because the syntax of the function calloc() is incorrect. In order to rectify this error, we must write: calloc(3,sizeof(int));

2. In the function malloc(), each byte of allocated space is initialized to zero.
a) True

b) False

ANSWER:- b
Explanation: In the function malloc(), allocated space is initialized to junk values. In calloc(), each byte of allocated space is initialized to zero.

3. Which of the following functions allocates multiple blocks of memory, each block of the same size?
a) malloc()
b) realloc()
c) calloc()
d) free()

ANSWER:- c
Explanation: malloc() allocates a single block of memory whereas calloc() allocates multiple blocks of memory, each block with the same size.

4. A condition where in memory is reserved dynamically but not accessible to any of the programs is called _____
a) Memory leak
b) Dangling pointer
c) Frozen memory
d) Pointer leak

ANSWER:- a
Explanation: If we allocate memory dynamically in a function (malloc, calloc, realloc), the allocated memory will not be de-allocated automatically when the control comes out of the function. This allocated memory cannot be accessed and hence cannot be used. This unused inaccessible memory results in a memory leak.

5. What will happens if the statement free(a) is removed in the following C code?

```
#include<stdio.h>
#include<stdlib.h>
main()
{
    int *a;
    a=(int*)malloc(sizeof(int));
    *a=100;
    printf("*a%d",*a);
    free(a);
    a=(int*)malloc(sizeof(int));
    *a=200;
    printf("a%p",a);
    *a=200;
    printf("a%d",*a);
}
```

a) Error
b) Memory leak
c) Dangling pointer arises
d) 200 is printed as output

ANSWER:- b
Explanation: The pointer ‘a’ points to the recent value 200, making the memory allocated earlier inaccessible. Hence, the memory where the value 100 is inaccessible and cannot be freed. Therefore, memory leak occurs.

6. The incorrect statement with respect to dangling pointers is _____
a) Pointer pointing to non-existent memory location is called dangling pointer
b) When a dynamically allocated pointer references the original memory after it has been freed, a dangling pointer arises
c) If memory leak occurs, it is mandatory that a dangling pointer arises
d) Dangling pointer may result in segmentation faults and potential security risks

ANSWER:- c
Explanation: Memory leak and dangling pointers are not inter dependent. Hence, when memory leak occurs, it is not mandatory that a dangling pointer arises

7. What will be the output of the following C code?

```
#include<stdio.h>
#include<stdlib.h>
void main()
{
    char *p = calloc(100, 1);
    p = "welcome";
    printf("%s\n", p);
}
```

a) error
b) welcome
c) memory location stored by the pointer
d) junk value

ANSWER:- b
Explanation: There is no error in the above code. The format specifier being %s, address is not returned. Hence, welcome is the output.

8. In the function realloc(), if the new size of the memory block is larger than the old size, then the added memory _____
a) is initialized to junk values
b) is initialized to zero
c) results in an error
d) is not initialized

ANSWER:- d
Explanation: The function realloc() changes the size of a particular memory block. If the new size

is larger than the old size, the added memory is not initialized.

9. The free() function frees the memory state pointed to by a pointer and returns _____
a) the same pointer
b) the memory address
c) no value
d) an integer value

ANSWER:- c
Explanation: The free() function frees the memory state pointed by a pointer and returns no value.

10. The following C code is an example of _____

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
main()
{
    char *p,*q;
    p=(char*)malloc(3*sizeof(char));
    q=p;
    strcpy(p,"hello");
    printf("p=%s",p);
    printf("q=%s",q);
    free(q);
    q=NULL;
    gets(p);
    gets(q);
    printf("%s",p);
    printf("%s",q);
}
```

a) Memory leak
b) Dangling pointer
c) Static memory allocation
d) Linked list

ANSWER:- b
Explanation: In the above code, the pointer p, points to a memory location which has been freed. Hece the above code is an example of dangling pointer.

1. What will be the output of the following C code if it is executed on a 32 bit processor?

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int *p;
    p = (int *)malloc(20);
    printf("%d\n", sizeof(p));
    free(p);
    return 0;
}
```

a) 2
b) 4
c) 8
d) Junk value

ANSWER:- b
Explanation: The size of a pointer is 2 bytes on a 16 bit platform, 4 bytes on a 32 bit platform and 8 bytes on a 64 bit platform.

2. The number of arguments taken as input which allocating memory dynamically using malloc() is _____
a) 0
b) 1
c) 2
d) 3

ANSWER:- b
Explanation: An example of memory allocated using malloc():
(int*)malloc(3*sizeof(int)
It is clear from the above example that malloc() takes only one argument as input, that is the number of bytes to be allocated.

3. Suppose we have a one dimensional array, named 'x', which contains 10 integers. Which of the following is the correct way to allocate memory dynamically to the array 'x' using malloc()?
a) x=(int*)malloc(10);
b) x=(int*)malloc(10,sizeof(int));
c) x=malloc(int 10,sizeof(int));
d) x=(int*)malloc(10*sizeof(int));

ANSWER:- d
Explanation: According to the syntax of malloc, the correct way to do the specified operation is: x=(int*)malloc(10*sizeof(int)); This operation can also be performed using calloc(). In that case, the method will be: x=(int*)calloc(10,sizeof(int));

4. What will be the error (if any) in the following C code?

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main()
{
    char *p;
    *p = (char)calloc(10);
    strcpy(p, "HELLO");
    printf("%s", p);
    free(p);
    return 0;
}
```

a) No error
b) Error in the statement: strcpy(p,”HELLO”);
c) Error in the statement: *p=(char)calloc(10);
d) Error in the statement: free(p);

ANSWER:- c
Explanation: The syntax for dynamically allocating memory using calloc() is incorrect. Hence, this code results in an error. The correct syntax for calloc is: void*calloc(size_t n, size_t size);

5. If malloc() and calloc() are not type casted, the default return type is _____
a) void*
b) void**
c) int*
d) char*

ANSWER:- a
Explanation: If malloc() and calloc() are not type casted, they return a pointer of the type void.

6. Pick out the correct statement with respect to the heap.
a) Local variables are stored on the heap
b) Static variables are stored on the heap
c) Heap is the data structure that is used to implement recursive function calls
d) Everything on the heap is anonymous

ANSWER:- d
Explanation: Local variables are stored on the stack. Static variables are stored in the permanent storage area. Stack is the data structure used to implement recursive function calls. Hence, it is true that everything on the heap is anonymous.

7. What will be the output of the following C code? (Given that the size of array is 4 and new size of array is 5)

```
#include<stdio.h>
#include<stdlib.h>
main()
{
    int *p,i,a,b;
    printf("Enter size of array");
    scanf("%d",&a);
    p=(int*)malloc(a*sizeof(int));
    for(i=0;i<a;i++)
    printf("%d\n",i);
    printf("Enter new size of array");
    scanf("%d",&b);
    realloc(p,b);
    for(i=0;i<b;i++)
    printf("%d\n",i);
    free(p);
}
```

a)

1234

12345

b) Error
c)

0123

01234

d)

0123

12345

ANSWER:- c
Explanation: In the above code, we are reallocating memory. When the size of the array is 4, the output is 0123. When the size of the array is changed to 5, the output is 01234. Hence, the output is:
0123
01234

8. When the pointer is NULL, then the function realloc is equivalent to the function _____
a) malloc
b) calloc
c) free
d) alloc

ANSWER:- a
Explanation: If pointer is NULL, the call to the function realloc is equal to malloc(size), for any value of size. If size is equal to zero, then the pointer is not NULL and the call is equivalent to free(pointer).

9. Garbage collector frees the programmer from worrying about _____
a) Dangling pointers
b) Creating new objects
c) Memory leak
d) Segmentation errors

ANSWER:- c
Explanation: A garbage collector is a program

that automatically removes unwanted data held temporarily in the memory during processing. Hence it frees the programmer from worrying about memory leaks.

10. If the space in memory allocated by malloc is not sufficient, then an allocation fails and returns _____
a) NULL pointer
b) Zero
c) Garbage value
d) The number of bytes available

ANSWER:- a
Explanation: A NULL pointer is returned when the memory allocated by malloc dynamically is insufficient.

1. The preprocessor directive used to give additional information to the compiler, beyond which is conveyed in the language _____
a) #include
b) #define
c) #pragma
d) #elif

ANSWER:- c
Explanation: The preprocessor directive #pragma is used to give additional information to the compiler, other than what is conveyed in the language itself.

2. What will be the output of the following C code, if it is run on a 32 bit platform?

```
#include<stdio.h>
#pragma(1)
struct test
{
    int i;
    char j;
};
main()
{
    printf("%d",sizeof(struct test));
}
```

a) Error
b) 1
c) 4
d) 8

ANSWER:- d
Explanation: #pragma pack(n), where n is the number of alignment in bytes. #pragma pack(1) is the directive for the compiler to pack the structure.

3. In the directive, #pragma pack(n), which of the following is not a valid value of n?
a) 1
b) 2
c) 3
d) 4

ANSWER:- c
Explanation: Valid arguments are 1,2,4 and 8. 3 is not a valid value for n.

4. Which of the following attributes is used to specify that the minimum required memory to be used to represent the types?
a) packed
b) aligned
c) unused
d) deprecated

ANSWER:- a
Explanation: The keyword __attribute__ allows you to specify special attributes of struct type. 6 attributes are currently defined for types: aligned, packed, transparent_union, unused, deprecated and may_alias. The attribute “packed” is used to specify that the minimum required memory to be used to represent the types.

5. In the directive #pragma pack(n), if the value of ‘n’ is given to be 5, then what happens?
a) Error
b) Warning but no error
c) Executes the pragma statement
d) Ignores the pragma statement and executes the program

ANSWER:- d
Explanation: Valid values for n are 1,2,4 and 8. If the value of n is one that the compiler does not recognize, then it simply ignores the pragma statement without throwing any error or warning.

6. The correct syntax of the attribute packed is _____
a) __attribute__((packed));
b) _attribute(packed);
c) _attribute_((packed));
d) __attribute__(packed);

ANSWER:- a
Explanation: The correct syntax of the attribute packed is: __attribute__((packed));

7. The pragma _____ is used to remove an identifier completely from a program.
a) GNU piston
b) GCC poison
c) GNU poison
d) GCC piston

ANSWER:- b
Explanation: There are several pragmas defines. One such pragma is GCC poison which is used to remove an identifier.

8. The function of __attribute__((packed)); can also be performed using _____
a) #pragma pack(1);
b) #pragma pack(2);
c) #pragma pack(4);
d) #pragma pack(8);

ANSWER:- a
Explanation: __attribute((packed)); and #pragma(1) are used to perform the same function, that is, to direct the compiler to pack the structure.

9. #pragma GCC poison should be followed by a list of identifiers that are _____
a) even in number
b) odd in number
c) valid
d) invalid

ANSWER:- d
Explanation: #pragma poison GCC is used to remove an identifier from a program. It should be followed by a list of identifiers which are not valid in the program, for example: scanf, printf etc.

10. What will be the output of the following C code?

```
#include<stdio.h>
#pragma GCC poison printf
main()
{
    printf("sanfoundry");
    return 0;
}
```

a) error is thrown
b) sanfoundry is printed
c) warning but no error
d) yrdnoufnas is printed

ANSWER:- a
Explanation: The code shown above results in an error: attempt to use poisoned printf
When the above program is compiled, it results in an error since #pragma was used to specify that the identifier printf should not be used in the program.

1. Which of the following is a stringizing operator?
a) < >
b) #
c) %
d) ##

ANSWER:- b
Explanation: # is the stringizing operator. It allows formal arguments within a macro definition to be converted to a string.

2. What will be the output of the following C code?

```
#define sanfoundry(s,n) #s #n
main()
{
    printf(sanfoundry(hello,world));
}
```

a) sanfoundry(hello,world)
b) sanfoundry
c) hello,world
d) helloworld

ANSWER:- d
Explanation: The output to this code will be helloworld because when we use the stringizing operator, the resulting string will automatically be concatenated (combined) with any adjacent strings.

2. What will be the output of the following C code?

```
#define sanfoundry(s,n) #s #n
main()
{
    printf(sanfoundry(hello,world));
}
```

a) sanfoundry(hello,world)
b) sanfoundry
c) hello,world
d) helloworld

ANSWER:- d
Explanation: The output to this code will be helloworld because when we use the stringizing operator, the resulting string will automatically be concatenated (combined) with any adjacent strings.

Check this: BCA MCQs | C Books

3. What will be the output of the following C code?

```
#define display(text) printf(#text "@")
main()
{
    display(hello.);
    display(good morning!);
}
```

a) hello.@good morning!
b) error
c) hello.good morning!@
d) hello.@good morning!@

ANSWER:- d
Explanation: Each actual argument is converted into string within the printf function. Each argument is concatenated with '@', which is written as a separate string within the macro definition.

4. What will be the output of the following C code?

```
#define display(a) #a
main()
{
    printf(display("56#7"));
}
```

a) Error
b) "56#7"
c) 56#7
d) 567

ANSWER:- b
Explanation: In this case, it is not necessary for the argument in the printf function to be enclosed in double quotes. However, if the argument is enclosed in double quotes, no error is thrown. The output of the code shown will be "56#7".

5. What will be the output of the following C code?

```
#define HELLO(a) #a
main()
{
    printf(HELLO(good        morning));
}
```

a) good morning
b) goodmorning
c) good morning
d) error

ANSWER:- c
Explanation: The output of the code shown above will be: good morning
In the resulting string, the consecutive blank spaces are replaced by a single blank space when we use the stringizing operator.

6. What will be the output of the following C code?

```
#include <stdio.h>
#define sanfoundry(x)  #x
int main()
{
    int marks=100;
    printf("value of %s is
= %d\n",sanfoundry(marks),marks);
    return 0;
}
```

a) error
b) value of marks=100
c) value of=100
d) 100

ANSWER:- b
Explanation: In the code shown above, the variable name(marks) is passed as an argument. By using the # operator, we can print the name of the variable as a string.

7. What will be the output of the following C code?

```
#define hello(c) #c
main()
{
    printf(hello(i,am));
}
```

a) i,am
b) iam
c) i am
d) error

ANSWER:- d
Explanation: The above code will result in an error. This is because we have passed to arguments to the macro hello, but it should be talking only one.

8. What will be the output of the following C code?

```
#define hello(c,d) #c #d
main()
{
    printf(hello(i,"am"));
}
```

a) iam
b) i"am"
c) am
d) "am"

ANSWER:- b
Explanation: The output for the following code will be i"am". Since 2 arguments are passed and the macro hello takes two arguments, there is no error.

9. What will be the output of the following C code?

```
#define F abc
#define B def
#define FB(arg) #arg
```

```
#define FB1(arg) FB(arg)
main()
{
    printf(FB(F B));
    FB1(F B);
}
```

a) F B
b) Error
c) FB
d) “FB”

ANSWER:- a
Explanation: The argument F B(only one space between F and B) is passed to the macro FB. This argument is converted to a string with the by the stringizing operator. Thus F B is printed.

10. What will be the output of the following C code?

```
#define display(text) "$" #text
main()
{
    printf(display(hello     world));
}
```

a) hello world
b) $helloworld
c) $hello world
d) error

ANSWER:- c
Explanation: The output of the code shown above is $hello world
The argument “hello world” is passed to the macro text. The symbol “$” is present from before. In the resulting string, all the blank spaces are replaced by a single blank space. In addition to this, “$” is concatenated to the beginning of the resultant string.

1. What will be the output of the following C code?

```
#include<stdio.h>
#define max 100
main()
{
    #ifdef max
    printf("hello");
}
```

a) 100
b) hello
c) “hello”
d) error

ANSWER:- d
Explanation: The code shown above results in an error. This is because the preprocessor #endif is missing in the above code. If the identifier is defined, then the statements following #ifdef are executed till #endif is encountered.

2. _____ is the preprocessor directive which is used to end the scope of #ifdef.
a) #elif
b) #ifndef
c) #endif
d) #if

ANSWER:- c
Explanation: The #ifdef preprocessor directive is used to check if a particular identifier is currently defined or not. If the particular identifier is defined, the statements following this preprocessor directive are executed till another preprocessor directive #endif is encountered. #endif is used to end the scope of #ifdef.

3. What will be the output of the following C code?

```
#include<stdio.h>
void main()
{
    #ifndef max
    printf("hello");
    #endif
    printf("hi");
}
```

a) hello
b) hellohi
c) error
d) hi

ANSWER:- b
Explanation: The code shown above illustrates the preprocessor directive #ifndef. If the identifier checked is not defined, then the statements following #ifndef are executed. Here, since the identifier max is not defined, the statement after #ifndef is executed. Hence the output is: hellohi

4. What will be the output of the following C code?

```
#include<stdio.h>
#define san 557
int main()
{
    #ifndef san
    printf("yes");
    #endif
    printf("no");
}
```

a) error
b) yes
c) no

d) yesno

ANSWER:- c
Explanation: The output to the above code will be no. Since we have used the preprocessor directive, #ifndef and defined the identifier san, “yes” is not printed. Hence only “no” is printed as output.

5. The preprocessor directive which checks whether a constant expression results in a zero or non-zero value _____
a) #if
b) #ifdef
c) #undef
d) #ifndef

ANSWER:- a
Explanation: #if checks whether a constant expression results in zero or not. If the expression is a non-zero value, then the statements between #if and #endif will be executed. If the constant expression results in a zero value, the statements between #if and #endif will not be executed.

6. What will be the output of the following C code?

```
#include<stdio.h>
#define max 100
void main()
{
    #if(max%10)
    printf("san");
    #endif
    printf("foundry");
}
```

a) error
b) san
c) foundry
d) sanfoundry

ANSWER:- d
Explanation: The code shown above is an illustration of the preprocessor directive #if. The value of the defined identifier max is 100. On checking the condition (100510==), which is true, the statements after #if are executed till #endif is encountered. Hence the output is sanfoundry.

7. The preprocessor directive which is used to remove the definition of an identifier which was previously defined with #define?
a) #ifdef
b) #undef
c) #ifndef
d) #def

ANSWER:- b
Explanation: #undef is used to remove the definition of any identifier which had been previously defined in the code with #define.

8. What will be the output of the following C code?

```
#include<stdio.h>
#define hello 10
void main()
{
    printf("%d",hello);
    #undef hello
    printf("%d",hello);
}
```

a) 10
b) hello
c) error
d) 1010

ANSWER:- c
Explanation: Error: hello undeclared. An error is thrown when the code shown above is executed because before the second call to the printf function, the macro hello was undefined using #undef.

9. What will be the output of the following C code?

```
#include <stdio.h>
#define a 2
main()
{
    int r;
    #define a 5
    r=a*2;
    printf("%d",r);
}
```

a) 10
b) 4
c) 2
d) 5

ANSWER:- a
Explanation: The macro ‘a’ is redefined in the code shown above. Hence the value of a, which is initially 2, is redefined to 5. The value stored in ‘r’ is the result of the expression (a*2), which is equal to 10. Hence the output of this code will be 10.

10. What will be the output of the following C code if the value of ‘p’ is 10 and that of ‘q’ is 15?

```
#include<stdio.h>
int main()
{
    int p,q;
    printf("Enter two numbers\n");
```

```
    scanf("%d",&p);
    scanf("%d",&q);
    #if(4<2)
    printf("%d",p);
    #elif(2>-1)
    printf("%d",q);
    #else
    printf("bye");
    #endif
}
```

a) 10
b) 15
c) bye
d) error

ANSWER:- b
Explanation: The output of the code shown above is 15. The values given for 'p' and 'q' are 10 and 15 respectively. Since the condition given for #elif is true, the value stored in 'q' will be printed, that is 15.

1. What will be the output of the following C code?

```
#include<stdio.h>
#define san 10
main()
{
    #ifdef san
    #define san 20
    #endif
    printf("%d",san);
}
```

a) 10
b) 20
c) Error
d) 1020

ANSWER:- b
Explanation: In the code shown above, if the identifier san is defined, then its value is redefined and changed to 20. It is then printed. Hence the output is 20.

2. What will be the output of the following C code?

```
#include<stdio.h>

#define hello
main()
{
    #ifdef hello
    #define hi 4
    #else
    #define hi 5
    #endif
    printf("%d",hi);

}
```

a) 4
b) 5
c) 45
d) error

ANSWER:- a
Explanation: The code shown above illustrates #define, #ifdef, #else, #endif. Since the identifier hello is defined (condition of #if is true), another identifier names hi is defined and it's value is 4. If hello was not defined, then the value of hi would be 5.

3. The purpose of the preprocessor directive #error is that _____
a) It rectifies any error present in the code
b) It rectifies only the first error which occurs in the code
c) It causes the preprocessor to report a fatal error
d) It causes the preprocessor to ignore an error

ANSWER:- c
Explanation: #error is a preprocessor directive which causes the preprocessor to report a fatal error. The tokens which form the rest of the line after #error are used as the error message.

4. What will be the output of the following C code?

```
#include<stdio.h>
#define max 20
main()
{
    #ifndef max
    #define min 10
    #else
    #define min 30
    #endif
    printf("%d",min);
}
```

a) 10
b) 20
c) 30
d) error

ANSWER:- c
Explanation: The output of the code shown above is 30. Since the identifier max is defined (the condition of #ifndef is true), hence another identifier, that is, min is defined and its value is equal to 30.

5. What will be the output of the following C code?

```
#include<stdio.h>
#define hello 10
main()
```

```
{
    #ifdef hello
    #undef hello
    #define hello 100
    #else
    #define hello 200
    #endif
    printf("%d",hello);
}
```

a) Error
b) 10
c) 100
d) 200

ANSWER:- c
Explanation: The output of the code shown above is 100. If the identifier hello is defined, then it is undefined and redefined. It's new value is 100. If the identifier was not defined, it would be defined with a value of 200.

6. What will be the output of the following C code?

```
#include<stdio.h>
#define sf 10
main()
{
    if(sf==100)
        printf("good");
    else
    {
        printf("bad");
        sf=100;
    }
    printf("%d",sf);
}
```

a) 100
b) bad
c) 10
d) error

ANSWER:- d
Explanation: The above code will result in an error because a macro cannot be defined using a simple assignment operator. In the code shown above, the value of sf is changed to 100 using the assignment operator. This causes an error.

7. What will be the output of the following C code?

```
#include<stdio.h>
void f()
{
    #define sf 100
    printf("%d",sf);
}
int main()
{
    #define sf 99;
    f();
    printf("%d",sf);
}
```

a) error
b) 100
c) 99
d) 10099

ANSWER:- a
Explanation: Macro definition is global even if it is appears within the function scope. In this case the macro sf is defined in the function f(). Hence it results in a compile time error.

8. What will be the output of the following C code?

```
#include<stdio.h>
#define INDIA 1
#define US 2
#define CC US
main()
{
    #if CC==INDIA
    printf("Rupee");
    #elif CC==US
    printf("Dollar");
    #else
    printf("Euro");
    #endif
}
```

a) Euro
b) Rupee
c) Dollar
d) Error

ANSWER:- c
Explanation: The output of the code shown above is Dollar. Since the macro CC is defined as US at the beginning of the code, it satisfies the condition #elif(CC==US). Hence "Dollar" is printed.

9. What will be the output of the following C code?

```
#define sqr(x) x*x
main()
{
    int a1;
    a1=25/sqr(5);
    printf("%d",a1);
}
```

a) 25
b) 1
c) 5
d) error

ANSWER:- a
Explanation: The output of the code shown above is 25. This is because the arithmetic statement takes the form of: 25/5*5 (which is equal to 1). If

the macro had been defined as #define sqr(x) (x*x), the output would have been 1.

10. Which of the following is not a preprocessor directive?
a) #error
b) #pragma
c) #if
d) #ifelse

ANSWER:- d
Explanation: #ifelse is not a preprocessor directive. #error, #pragma, #if are preprocessor directives. There is a preprocessor directive, #elif, which performs the function of else-if.

1. Which of the following operators is used to concatenate two strings without space?
a) #
b) < >
c) **
d) ##

ANSWER:- d
Explanation: The operator ## is used for token concatenation (to concatenate two strings without space).

2. What will be the output of the following C code?

```
#include <stdio.h>
#define p( n ) printf( "t" #n " = %d", t##n )
int t3=10;
int main()
{
   p(3);
}
```

a) t=10
b) t3=10
c) t10=3
d) t=3

ANSWER:- b
Explanation: The code shown above uses the ## operator to concatenate 't' and 3. t3 has been declared as 10 in the code, Hence the output of the code shown above is: t3=10

3. What will be the output of the following C code?

```
#include <stdio.h>
#define p( n,m ) printf( "%d", m##n )
int main()
{
   p(3,4);
}
```

a) Error
b) Junk value
c) 34
d) 43

ANSWER:- d
Explanation: In the code shown above, we have concatenated the two tokens in the order: mn. The value of m is equal to 4 and that of n is equal to 3. Hence the output of this code is 43.

4. What will be the output of the following C code?

```
#include <stdio.h>
#define p( n,m ) printf( "%d", m##n )
#define q(a,b) printf("%d",a##b)
main()
{
   p(3,4);
   q(5,5);
}
```

a) 4356
b) 3456
c) 4365
d) 3465

ANSWER:- a
Explanation: In the code shown above we have concatenated m and n in one statement and a and b in another. Since we have not used a new line character, the output of this code will be equal to 4356.

5. The following C code results in an error.

```
#include <stdio.h>
#define world( n ) printf( "t^^" #n"
= %c", t##n )
int t3=1;
int main()
{
   world(3);
}
```

a) True
b) False

ANSWER:- b
Explanation: The code shown above does not result in an error. The output of this code is t^^3=junk value.

6. What will be the output of the following C code?

```
#include <stdio.h>
#define display( n ) printf( "a" #n "
= %d", a##n )
int main()
{
   display(3);
```

```
}
```

a) a3
b) 31
c) a 3
d) error

ANSWER:- d
Explanation: The code shown above results in an error because we have not explicitly declared a3. Had we declared a3 in this code, it would not have thrown an error.

7. What will be the output of the following C code?

```
#include <stdio.h>
#define hello( n ) a##n
int a3;
int main()
{
    int x;
    x=hello(3);
    if(x!=0)
        printf("hi");
    else
        printf("good");
}
```

a) error
b) a3
c) good
d) hi

ANSWER:- c
Explanation: The code shown will print 'hi' if x is not equal to zero and 'good' if x is equal to zero. In the above code, we have declared x (equal to zero). Hence 'good' is printed as output.

8. What will be the output of the following C code?

```
#include <stdio.h>
#define hello( n ) printf( "a" #n
"= %d", a##n )
int a3=3;
int main()
{
    #ifdef a3
       hello(3);
   #else
       printf("sorry");
   #endif
}
```

a) a3=3
b) error
c) a=3
d) sorry

ANSWER:- d
Explanation: The code shown above prints a3=3 if as is defined. Since a3 is not defines, 'sorry' is printed as output.

9. What will be the output of the following C code?

```
#include <stdio.h>
#define p( n ) printf( "t*" #n " = %s",
t##n )
char tsan[]="tsan";
int main()
{
    int x;
    x=p(san);
}
```

a) error
b) tsan=tsan
c) t*san=t*san
d) t*san=tsan

ANSWER:- d
Explanation: The code shown above uses the ## operator to concatenate the tokens t* and san. We have decalred tsan[]=tsan. Hence the output of the code shown will be: t*san=tsan.

10. What will be the output of the following C code?

```
#include <stdio.h>
#define p( n ) printf( "t%\n" #n "
= %d", t##n )
int t3=10;
int main()
{
    int x;
    x=p(3);
}
```

a) t3=10
b) t3=10
c) t%3=10
d) t%3=10

ANSWER:- a
Explanation: In this code, operator ## is used to concatenate t and 3. However, a new line character is a part of this statement. The variable t3 has been declared as 10. Hence the output of this code will be: t
3=10

1. Name the function whose definition can be substituted at a place where its function call is made _____
a) friends function
b) inline function
c) volatile function
d) external function

ANSWER:- b
Explanation: The inline function, whose definitions being small can be substituted at a place where its function call is made. They are inlined with their function calls.

2. What will be the output of the following C code?

```
#include <stdio.h>
void inline func1(int a, int b)
{
    printf ("a=%d and b=%d\n", a, b);
}
int inline func2(int x)
{
    return x*x;
}
int main()
{
    int tmp;
    func1(1,4);
    tmp = func2(6);
    printf("square val=%d\n", tmp);
    return 0;
}
```

a) a=1 and b=4
square val = 36
b) a=4 and b=1
c) error
d) square val = 36

ANSWER:- a
Explanation: The code shown above is an example of inline function. Both functions 1 and 2 are inline functions. Hence the output will be as shown in option
a=1 and b=4

square val = 36

3. What will be the error (if any) in the following C code?

```
#include <stdio.h>
void inline func1(float b)
{
    printf ("%lf\n",b*2);
}
int main()
{
     inline func1(2.2);
     return 0;
}
```

a) No error
b) Error in statement: void inline func1(float b)
c) Error in statement: printf("%lf\n",b*2);
d) Error in statement: inline func1(2.2);

ANSWER:- d
Explanation: The keyword inline is used only with the function definition. In the code shown above it has been used with the function call. Hence there is an error in the statement: inline func1(2.2);

4. What will be the output of the following C code?

```
#include <stdio.h>
void inline f1(char b)
{
    printf ("%d\n",b);
}
int main()
{
    f1('a');
    return 0;
}
```

a) a
b) 65
c) error
d) 97

ANSWER:- d
Explanation: The definition of the function f1 is replaced with the function call. Since the format specifier used is %d, the ASCII value of the said character is printed. Hence the output of the code shown above is 97 (ASCII value of the alphabet 'a').

5. What will be the output of the following C code?

```
#include <stdio.h>
void inline func1(char b[10])
{
    printf ("%c\n",b[2]);
}
int main()
{
     func1("sanfoundry");
     return 0;
}
```

a) s
b) n
c) a
d) error

ANSWER:- b
Explanation: In the code shown above, the definition of function func1 is replaced with its function call. The function prints the alphabet at the second index of the given string. Hence the output of this code is 'n'.

6. The following C code results in an error. State whether this statement is true or false.

```
#include <stdio.h>
void f(double b)
{
```

```
    printf ("%ld\n",b);
}
int main()
{
     inline f(100.56);
     return 0;
}
```

a) True
b) False

ANSWER:- a
Explanation: The above code will result in an error because we have used the keyword inline in the function call. Had the keyword inline been used in the function definition, the code would not have thrown an error.

7. What will be the output of the following C code?

```
#include<stdio.h>
static inline int max(int a, int b)
{
  return a > b ? a : b;
}
main()
{
    int m;
    m=max(-6,-5);
    printf("%d",m);
}
```

a) -6
b) -5
c) Junk value
d) Error

ANSWER:- b
Explanation: The code shown a replaces the definition of a function max with it's function call. This function is used to print the bigger of the two arguments. Hence -5 is printed.

8. What will be the output of the following C code?

```
#include<stdio.h>
#define inline
inline f(char a)
    {
        #ifdef inline
        printf("%c",a);
        #endif
    }
main()
{
    f('a');
}
```

a) Error
b) a
c) No error but nothing will be printed as output
d) 97

ANSWER:- b
Explanation: In the code shown above, we have defined identifier names inline using the macro #define. The output will be printed only if inline is defined. Since it is defined, 'a' is printed as output.

9. What will be the output of the following C code?

```
#include<stdio.h>
extern inline int min(int a, int b)
{
  return a < b ? a : b;
}
main()
{
    int m;
    m=min(3,-5);
    printf("%d",m);
}
```

a) Error
b) 3
c) -5
d) Junk value

ANSWER:- a
Explanation: The above code will result in an error due to the statement: extern inline int min(int a, int b). The keyword 'extern' causes an error: undefined reference to min.

10. To have GCC inline the given function regardless of the level of optimization, we must declare the function with the attribute _____

a) optimize_inline
b) packed_inline
c) always_inline
d) level_inline

ANSWER:- b
Explanation: To have GCC inline the given function regardless of the level of optimization, we declare the function with the attribute always_inline.

1. A machine in which the least significant byte is stored in the smallest address is _____
a) Big endian machine
b) Bi-endian machine
c) Binary bit machine
d) Little endian machine

ANSWER:- d
Explanation: In little endian machines, last byte of

binary representation (least significant byte) of a multi byte data type is stored first, whereas in big endian method, the first byte of binary representation of a multi byte data type is stored first.

2. If the output of the following C code is “Big endian”, then what will be the value of *a is?

```
#include <stdio.h>
int main()
{
   unsigned int i = 1;
   char *a = (char*)&i;
   if (*a)
       printf("Little endian");
   else
       printf("Big endian");
   getchar();
   return 0;
}
```

a) -1
b) 0
c) 1
d) 2

ANSWER:- b
Explanation: In the code shown above, a character pointer ‘a’ points to an integer ‘i’. Since the size of the character is 1 byte, when the character pointer is de-referenced, it contains 1 integer byte. If the machine is a little endian machine then the value of *a will be 1, since the last byte is stored first. If the machine is big endian, the value of *a will be 0.

3. It is possible for a processor to support both little and big endian methods.
a) True
b) False

ANSWER:- a
Explanation: It is possible for a processor to support both little and big endian methods. Such machines are known as bi-endian machines.

4. The standard byte order for networks is _____
a) Bit-Binary
b) Little endian
c) Big endian
d) Bi-endian

ANSWER:- c
Explanation: The standard byte order (network byte order) for networks is big endian. Before transferring data on a network, data is converted to big endian.

5. Which of the following is not an example of big endian machines?
a) Power PC
b) Motorola 68K
c) SPARC processors
d) ARM processors

ANSWER:- d
Explanation: ARM processor is an example of little endian machine. Current generation ARM processor is an example of bi-endian machine. Power PC, Motorola 68K and SPARC are examples of big endian mahines.

6. Suppose we transfer a file written on a little endian machine to a big endian machine and there is no transformation from little endian to big endian, then what happens?
a) The big endian machine throws an error when it receives the file
b) The big endian machine reads the file in the reverse order
c) The big endian machine does not read the file
d) The big endian machine reads the file in the normal order

ANSWER:- b
Explanation: If there is no transformation from little endian to big endian on transfer of a file from a little endian machine to a big endian machine, the big endian machine reads the file in reverse order.

7. File formats which have _____ as a basic unit are independent of endianness.
a) 1 byte
b) 2 bytes
c) 3 bytes
d) 4 bytes

ANSWER:- a
Explanation: File formats which have 1 byte as the basic unit are not endianness dependent. Eg: ASCII files. Other file formats have fixed endianness formats.

8. If the code shown below is executed on a little endian machine, then what will be the output of the following C code?

```
#include<stdio.h>
main()
{
    int y=1;
    printf("%d", (*(char*)&y));
}
```

a) 1
b) 1000
c) 9999
d) 0

ANSWER:- a
Explanation: The output of the above code will be one when it is run on a little endian machine. This is because a little endian machine stores the least significant byte in the smallest address.

9. If the data “ABCD” is to be stored in a little endian machine, it will be stored as _____
a) ABCD
b) DCBA
c) CDAB
d) BCDA

ANSWER:- b
Explanation: In a little endian machine, the least significant byte (right most) is stored in the smallest address. Hence the data “ABCD” will be stored in the form of “DCBA”.

10. The sequential order used to interpret a range of bytes in the memory of a computer is known as _____
a) Ordering
b) Sequencing
c) Endianness
d) Byte prediction

ANSWER:- c
Explanation: Endianness is the sequential order used to interpret a range of bytes in the memory of a system.

1. What will be the output of the following C code?

```
#include<stdio.h>
main()
{
    int n;
    n=f1(4);
    printf("%d",n);
}
f1(int x)
{
    int b;
    if(x==1)
        return 1;
    else
        b=x*f1(x-1);
        return b;
}
```

a) 24
b) 4
c) 12
d) 10

ANSWER:- a
Explanation: The above code returns the factorial of a given number using the method of recursion. The given number is 4 in the above code, hence the factorial of 4, that is, 24 will be returned.

2. The data structure used to implement recursive function calls _____
a) Array
b) Linked list
c) Binary tree
d) Stack

ANSWER:- d
Explanation: The compiler uses the data type stack for implementing normal as well as recursive function calls.

3. The principle of stack is _____
a) First in first out
b) First in last out
c) Last in first out
d) Last in last out

ANSWER:- c
Explanation: A stack is a last in first out(LIFO) data type. This means that the last item to get stored in the stack is the first item to get out of it.

4. In the absence of a exit condition in a recursive function, the following error is given _____
a) Compile time error
b) Run time error
c) Logical error
d) No error

ANSWER:- b
Explanation: When a recursive function is called in the absence of an exit condition, it results in an infinite loop due to which the stack keeps getting filled(stack overflow). This results in a run time error.

5. What will be the output of the following C code?

```
#include<stdio.h>
main()
{
    int n,i;
    n=f(6);
    printf("%d",n);
}
f(int x)
{
    if(x==2)
            return 2;
    else
    {
```

```
        printf("+");
        f(x-1);
    }
}
```

a) ++++2
b) +++++2
c) +++++
d) 2

ANSWER:- a
Explanation:
When x=6: '+' is printed.
When x=5: '+' is printed.
When x=4: '+' is printed.
When x=3: '+' is printed.
When x=2: 2 is printed.
Hence the output is: ++++2.

6. How many times is 'a' printed when the following C code is executed?

```
#include<stdio.h>
main()
{
    int a;
    a=f1(10);
    printf("%d",a);
}
f1(int b)
{
    if(b==0)
        return 0;
    else
    {
        printf("a");
        f1(b--);
    }
}
```

a) 9 times
b) 10 times
c) 0 times
d) Infinite number of times

ANSWER:- d
Explanation: Although we have specified the exit condition, the code above results in an infinite loop because we have used b- -(decrement operator) to call the recursive function. Due to this, the loop goes on infinitely. However, if we had used f1(b-1) instead, the answer would have been 10 times.

7. What will be the output of the following C code?

```
#include<stdio.h>
int main()
{
    int n=10;
    int f(int n);
    printf("%d",f(n));
}
int f(int n)
{
    if(n>0)
        return(n+f(n-2));
}
```

a) 10
b) 80
c) 30
d) Error

ANSWER:- c
Explanation: The recursive function returns n+f(n-2) till 10>0.
Therefore, the above code will be evaluated as: 10+8+6+4+2, which is equal to 30.

8. What will be the output of the following C code?

```
#include<stdio.h>
int main()
{
    printf("Hello");
    main();
    return 0;
}
```

a) Hello is printed once
b) Hello infinite number of times
c) Hello is not printed at all
d) 0 is returned

ANSWER:- b
Explanation: in the above code, we are calling main() from main(), which is recursion. However, we have not defined any condition for the program to exit. Hence, "hello" will be printed infinite number of times. To prevent this, we need to define a proper exit condition in the recursive function.

9. What will be the output of the following C code if the input given to the code shown below is "sanfoundry"?

```
#include<stdio.h>
#define NL '\n'
main()
{
    void f(void);
    printf("enter the word\n");
    f();
}
void f(void)
{
    char c;
    if((c=getchar())!=NL)
{
f();
    printf("%c",c);
}
```

```
return;
}
```

a) sanfoundry
b) infinite loop
c) yrdnuofnas
d) fnasyrdnuo

ANSWER:- c
Explanation: The above code prints the reverse of the word entered. The recursive function terminates when getchar() is equal to null.

10. Iteration requires more system memory than recursion.
a) True
b) False

ANSWER:- b
Explanation: Recursion requires more system memory than iteration due to the maintenance of stack.

1. In a signed integer, the sign is represented by _____
a) Least significant bit
b) Most significant bit
c) System dependent
d) The mean of the most significant bit and the least significant bit

ANSWER:- b
Explanation: In a signed integer, the most significant bit represents the sign whereas, in an unsigned integer, no bit is used to represent the sign.

2. Sign qualifiers can be applied to double.
a) True
b) False
ANSWER:- b
Explanation: Sign qualifiers cannot be applied to the data types: float and double
3. What will be the output of the following C code?

```
#include<stdio.h>
int main()
{
    signed char ch= 'a';
    printf("%u",ch);
    return 0;
}
```

a) -65
b) 65
c) -97
d) 97

ANSWER:- d
Explanation: Only for completeness, sign qualifiers are available with char data type.

4. What will be the output of the following C code?

```
#include<stdio.h>
main()
{
    signed char a[]= "BAT";
    printf("%c", a[1]);
    return 0;
}
```

a) -A
b) BAT
c) A
d) 65
ANSWER:-- c
Explanation: Sign qualifiers are available for char data types only for completeness. Format specifier used: %c, which prints a character. a[1] will print 'A'.
5. What is the default state of an integer in c language?
a) Signed
b) Unsigned
c) System dependent
d) There is no default state

ANSWER:- a
Explanation: The default state of an integer in c language is signed. Hence we can store both positive and negative integer values in it.

6. Suppose we have: int a=100; (signed by default).
If we want to convert this to an unsigned long integer, we can do this by making the following small change:
a) int a=lu100;
b) int a= 100ul;
c) int a=uu100;
d) int a=100uu;
ANSWER:- b
Explanation: An integer literal can have a suffix that is a combination of U and L(uppercase or lowercase). The prefix specifies the base or the radix, ie: 0x for hexadecimal, 0 for octal and nothing for decimal.
7. What is the binary representation of the integer -14?
a) 11110
b) 01110
c) 01100
d) 11100
ANSWER:- a
Explanation: The left most bit (most significant bit)

is used to represent the sign of the number: 0 for positive and 1 for negative. For example, a value of positive 14 is written as 01110(in binary). But a value of negative 14 is written as: 11110.

8. Which of the following header files must necessarily be included in your code, if you want to find the minimum value of unsigned short integer?
a) stdio.h
b) stdlib.h
c) limits.h
d) math.h
ANSWER:- c
Explanation: The header file limits.h is used to find the value of constants such as minimum and maximum. stdio.h: standard input and output, stdlib.h: standard library, math.h: mathematics functions library
9. What will be the error in the following C code?

```
main()
{
    long float a=-25.373e22;
    printf("%lf",a);
}
```

a) Negative number cannot be assigned to float data type
b) Long and float cannot be used together
c) Does not result in error
d) Logical error

ANSWER:- b
Explanation: It is not possible to use sign qualifier with float data type, however in the above code a negative value has been assigned to the variable without the use of sign qualifiers, which is possible. Hence, the answer is not “negative number cannot be assigned to float data type”. we know that size qualifier long is not applicable for float data type, hence the error.

10. What will be the output of the following C code?

```
main()
{
    unsigned a=10;
    long unsigned b=5l;
    printf("%lu%u",a,b);
}
```

a) 105
b) 510
c) 10
d) error

ANSWER:- a
Explanation: The format specifiers which are for unsigned and long unsigned have been used in the code. Had we used the format specifiers in the reverse order, the output would still be the same.

1. Determine the output of the C code mentioned below:

#include <stdio.h>

int main()

{

float q = ‘a’;

printf(“%f”, q);

return 0;

}

a. run time error

b. a

c. 97.000000

d. a.0000000

ANSWER:- (c) 97.000000

2. Which of these is NOT a relational or logical operator?

a. =

b. ||

c. ==

d. !=

ANSWER:- (a) =

3. What will be the output of the following C code?

#include<stdio.h>

int main()

{

int p = 1, q = 2, r = 3, s = 4, x;

e = r + s = q * p;

printf(“%d, %d\n”, x, s);

}

a. Syntax error

b. 5, 2

c. 7, 2

d. 7, 4

ANSWER:- (a) Syntax error

4. Out of the following, which one is not valid as an if-else statement?

a. if ((char) x){}

b. if (x){}

c. if (func1 (x)){}

d. if (if (x == 1)){}

ANSWER:- (d) if (if (x == 1)){}

5. We cannot use the keyword 'break' simply within _____.

a. while

b. for

c. if-else

d. do-while

ANSWER:- (c) if-else

6. The output of the C code mentioned below would be:

```
#include <stdio.h>
int main()
{
int a = 0, b = 0;
l1: while (a < 2)
{
a++;
while (b < 3)
{
printf("find\n");
goto l1;
}
}
}
```

a. loop loop find loop

b. Infinite error

c. find loop

d. Compile time loop

ANSWER:- (c) find loop

7. The global variables are _____.

a. External

b. Internal

c. Both External and Internal

d. None of the above

ANSWER:- (a) External

8. Out of the following operations, which one is not possible in the case of a register variable?

a. Global declaration of the register variable

b. Copying the value from the memory variable

c. Reading any value into the register variable

d. All of the above

ANSWER:- (d) All of the above

9. The #include <stdio.h> is a _____.

a. Inclusion directive

b. File inclusion directive

c. Preprocessor directive

d. None of the above

ANSWER:- (c) Preprocessor directive

10. Which of these properties of #define is false?

a. These always obey the scope rules

b. We can make use of a pointer to #define

c. The #define can be externally available

d. All of the above

ANSWER:- (d) All of the above

11. The correct format of declaring a function is:

a. type_of_return name_of_function (argument type);

b. type_of_return name_of_function (argument type){}

c. type_of_return (argument type) name_of_function;

d. all of the above

ANSWER:- (a) type_of_return name_of_function (argument type);

12. Out of the following function definition, which one will run correctly?

a.

```
int sum(int x, int y)
return (x + y);
```

b.

```
int sum(int x, int y)
{return (x + y);}
```

c.

```
int sum(x, y)
return (x + y);
```

d. none of the above

ANSWER:- (b)

int sum(int x, int y)

{return (x + y);}

13. Comment on the C statement given below:

int (*p)[5];

a. A ragged array

b. An array “p” of pointers

c. A pointer “p” to an array

d. None of the above

ANSWER:- (c) A pointer “p” to an array

14. The output of the C code mentioned below would be:

#include <stdio.h>

struct employee

{

int id;

char rank[5];

}

void main()

{

struct employee e;

s.no = 30;

printf(“howdy”);

}

a. hello

b. Compile-time error

c. 5, 30

d. Varies

ANSWER:- (b) Compile-time error

15. We can test the presence of a loop in a linked list by _____.

a. Comparing the node’s address with the address of all the other nodes

b. Travelling the list. In case we encounter the NULL, then no loop exists

c. Comparing the stored values of a node with the values present in all the other nodes

d. None of the above

ANSWER:- (a) Comparing the node’s address with the address of all the other nodes

16. Determine what’s true for x, if we define x as “int *x[10];”?

a. This definition will only allocate 10 pointers but will not initialize them

b. The initialization must be explicitly performed

c. The definition will only allocate 10 pointers but will not initialize them. Also, the initialization has to be explicitly performed

d. Error

ANSWER:- (c) The definition will only allocate 10 pointers but will not initialize them. Also, the initialization has to be explicitly performed

17. Which of these is the correct initialization method for the following:

typedef char *string;

a. More than a single space shouldn’t be given when we are using typedef

b. *string p = ‘A’;

c. string p = “Hello”;

d. *string *p = “Hello”;

ANSWER:- (c) string p = “Hello”;

18. We can determine the size of a union with the help of the size of _____.

a. The sum of all the members’ sizes

b. The biggest member of the union

c. The last member of the union

d. The first member of the union

ANSWER:- (b) The biggest member of the union

19. How much percentage of memory will be saved if we use bit-fields for the given C structure as compared to when we don’t use bit-fields for this very structure? (Assume the size of int to be 4).

```
struct temp
{
int m : 1;
int n : 2;
int o : 4;
int p : 4;
}s;
```

a. 33.3%

b. 75%

c. 25%

d. 50%

ANSWER:- (b) 75%

20. Out of the following snippet, which one will generate random numbers effectively?

a. rand(time(NULL));

b. rand(10);

c. rand();

d. all of the above

ANSWER:- (c) rand();

21. We can achieve the modulus for float by:

a. x % y

b. modulus(x, y);

c. fmod(x, y);

d. mod(x, y);

ANSWER:- (c) fmod(x, y);

22. _____ tells a compiler that the data would be defined somewhere and it would be connected to the linker.

a) variable

b) yvals

c) errno

d) extern

ANSWER:- (d) extern

23. The definition of the function abort() is in which header file?

a) stdlib.h

b) assert.h

c) stdio.h

d) stdarg.h

ANSWER:- (a) stdlib.h

24. What is the primary purpose of the preprocessor directive #error?

a) Rectifies the first error occuring in the code

b) Rectifies the errors present in a code

c) Causes a preprocessor to ignore any error

d) Causes a preprocessor to report some fatal error

ANSWER:- (d) Causes a preprocessor to report some fatal error

25. _____ is a condition in which the memory is dynamically reserved but isn't accessible to any program.

a) Pointer Leak

b) Frozen Memory

c) Dangling Pointer

d) Memory Leak

ANSWER:- (a) Memory Leak

3. C language is a successor to which language?

A. Basic
B. Cobol
C. C++
D. B

ANSWER:- D) B

Explanation:C programming language is a successor to the programming language B.

4. C is a ___.

A. Low level language
B. High level language
C. Medium level language
D. None of the above

ANSWER:- C) Medium level language

Explanation:C is a medium-level language because it contains the features of low-level language as well as high-level language.

6. C language is a ___.

A. Procedural oriented programming language
B. General purpose programming language
C. Structured programming
D. All of the above

ANSWER:- D) All of the above

Explanation: programming language is a general-purpose, procedural computer programming language that supports structured programming also.

7. Which is not a valid keyword in C language?

A. for
B. while
C. do-while
D. switch

ANSWER:- C) do-while

Explanation:do-while is not a valid keyword in the C programming language. It's a control statement. 'do' and 'while' are the separate keywords. The rest of all 'for', 'while', and 'switch' are the valid keywords in C.

8. What is an identifier in C language?

A. An identifier is a combination of alphanumeric characters used for conditional and control statements
B. An identifier is a combination of alphanumeric characters used for any variable, function, label name
C. Both A and B
D. None of the above

ANSWER:- B) An identifier is a combination of alphanumeric characters used for any variable, function, label name

Explanation: An identifier is a combination of alphanumeric characters used for any variable, function, label name. An identifier is a name that is used to identify the variables/ constants, functions, arrays, label name, and user-defined data.

9. A C-style comment, simply surround the text with ___.

A. /* and */
B. // and //
C. //
D. /** and **/

ANSWER:- A) /* and */

Explanation:A C-style comment, simply surround the text with /* and */.

10. Can we place comments between the statement to comments a part of the code?

A. Yes
B. No

ANSWER:- A) Yes

Explanation: Yes, we can place comments between the statement to comments a part of the code.

Example:

```
printf(/*"Hello World"*/ "Hey, how are you?");
```

11. ___ is an informal name for ISO/IEC 9899:1999, a past version of the C programming language standard?

A. C
B. C++
C. C89
D. C99

ANSWER:- D) C99

Explanation:C99 is an informal name for ISO/IEC 9899:1999, a past version of the C programming language standard.

12. In which version of C language, the C++ Style comment (//) are introduced?

A. C17
B. C18
C. C89
D. C99

ANSWER:- D) C99

Explanation:C language version C99 introduced C++ style comment (//), they can be used to comment a single line.

13. The C source file is processed by the ___.

A. Interpreter
B. Compiler
C. Both Interpreter and Compiler
D. Assembler

ANSWER:- B) Compiler

Explanation: The C source file is processed by the compiler.

14. How many whitespace characters are allowed in C language?

A. 2
B. 3
C. 4
D. 5

ANSWER:- D) 5

Explanation: There are 5 whitespace characters are allowed in C language, they are:

i. Space

ii. Horizontal tab
iii. Vertical tab
iv. Form feed
v. New-line

15. How many punctuation characters are allowed in C language?

A. 29
B. 30
C. 31
D. 32

ANSWER:- A) 29

Explanation: There are 29 punctuation characters are allowed in C language, they are:

```
_ { } [ ] # ( ) < > % : ; . ? * + - /
^ & | ~ ! = , \ " '
```

16. What is the extension of a C language source file?

A. .c
B. .cpp
C. .c99
D. .h

ANSWER:- A) .c

Explanation: The extension of a C language source file is ".c".

17. What is the extension of a C language header file?

A. .c
B. .cpp
C. .c99
D. .h

ANSWER:- D) .h

Explanation: The extension of a C language source file is ".h".

18. To develop which operating, C language was invented?

A. Linux
B. Unix
C. Android
D. Mac

ANSWER:- B) Unix

Explanation:C language was invented to develop Unix operating system.

19. Does C language support object-oriented approach?

A. Yes
B. No

ANSWER:- B) No

Explanation:C language does not support object-oriented approach.

20. Which is/are the disadvantage(s) of C language?

A. No Garbage Collection
B. Inefficient Memory Management
C. Low level of abstraction
D. Lack of Object Orientation
E. All of the above

ANSWER:- E) All of the above

Explanation:

The main disadvantages of C language are:

i. No Garbage Collection
ii. Inefficient Memory Management
iii. Low level of abstraction
iv. Lack of Object Orientation

21. Which are the fundamental data types in C?

A. char
B. int
C. float
D. All of the above

ANSWER:- D) All of the above

Explanation: The fundamental / basic data types in C language:

- char
- int
- float

22. How many byte(s) does a char type take in C?

A. 1

B. 2
C. 3
D. 4

ANSWER:- A) 1

Explanation: The `char` data type takes one byte in the memory.

23. For which type, the format specifier "%i" is used?

A. int
B. char
C. float
D. double

ANSWER:- A) int

Explanation: In C programming language, both of the format specifier `%d` and `%i` are used for `int` type, where `%d` specifies the type of variable as decimal and `%i` specifies the type as integer.

24. What is the difference between float and double in C?

A. both are used for the same purpose
B. double can store just double value as compare to float value
C. double is an enhanced version of float and was introduced in C99
D. double is more precise than float and can store 64 bits

ANSWER:- D) double is more precise than float and can store 64 bits

Explanation: In C programming language, the `double` is more precise than `float` and can store 64 bits.

25. Which is the correct format specifier for double type value in C?

A. %d
B. %f
C. %lf
D. %LF

ANSWER:- C) %lf

Explanation: The `%lf` is used to represent a double type value in C programming language.

26. The short type represents ___.

A. int
B. float
C. unsigned int
D. short int

ANSWER:- C) unsigned int

Explanation: The `short` type represents `short int` in C language.

27. How many byte(s) does a short type take in C?

A. 1
B. 2
C. 3
D. 4

ANSWER:- B) 2

Explanation:

In C programming language, the `short` or `short int` takes 2 bytes (16 bits) in memory.

28. What is the correct syntax to declare a variable in C?

A. data_type variable_name;
B. data_type as variable_name;
C. variable_name data_type;
D. variable_name as data_type;

ANSWER:- A) data_type variable_name;

Explanation: In C language, the correct syntax to declare a variable is:

```
data_type variable_name;
```

Where, *data_type* is the type of data (such as *int*, *char*, *float*, etc) and *variable_name* is a valid identifier. For example: *int age;*

29. How many types of qualifiers are there in C language?

A. 2
B. 3
C. 4
D. 5

ANSWER:- B) 3

Explanation: There are 3 types of qualifiers in C language, they are:

- Size qualifiers
- Sign qualifiers
- Type qualifiers

30. Which is/are the size qualifier(s) in C language?

A. short
B. long
C. double
D. Both A. and B

ANSWER:- D) Both A. and B.

Explanation: The size qualifiers are `short` and `long`.

31. Which is/are the sign qualifier(s) in C language?

A. signed
B. unsigned
C. long
D. Both A. and B

ANSWER:- D) Both A. and B.

Explanation: The sign qualifiers are used to specify the signed nature of an integer type. The sign qualifiers are `signed` and `unsigned`.

32. Which is/are the type qualifier(s) in C language?

A. const
B. volatile
C. static
D. Both A. and B

ANSWER:- D) Both A. and B.

Explanation:

The type qualifiers are const and "volatile".

33. Which is correct with respect to the size of the data types in C?

A. char > int > float
B. char < int < float
C. int < char < float
D. int < chat > float

ANSWER:- B) char < int < float

Explanation: The correct order of the data types as per the size is: `char < int > float`.

34. Which operator is used to find the remainder of two numbers in C?

A. /
B. \
C. %
D. //

ANSWER:- C) %

Explanation:

The `%` operator is known as "Modulus Operator" and it is used to find the remainder of two numbers.

35. Which of the following is not an arithmetic expression?

A. x = 10
B. x /= 10
C. x %= 10
D. x != 10

ANSWER:- D) x != 10

Explanation:`x != 10` is not a valid arithmetic expression.

36. What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
    int x = 20;
    x %= 3;
    printf("%d",x);

    return 0;
}
```

A. 2
B. 2.5
C. Error
D. Warning

ANSWER:- A) 2

Explanation: In the above code, the value of `x` is 20 and then in the next statement, the expression is `x %= 3`. That will be evaluate as:

```
x %= 3;
x = x % 3;
x = 20 %3;
x = 2
```

37. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    float x = 21.0;
    x %= 3.0;
    printf("%f",x);

    return 0;
}
```

A. 7
B. 7.00
C. 7.000000
D. Error

ANSWER:- D) Error

Explanation: In the above code, we are performing modulus operation with `float` values. Modulus operator doesn't work with `float` and `double` operands. Thus, the output will be:

```
error: invalid operands to binary %
(have ‘float’ and ‘double’)
```

38. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    float x = 23.456;
    printf("%.2f",x);
    return 0;
}
```

A. 23.45600
B. 23.456
C. 23.45
D. 23.46

ANSWER:- D) 23.46

Explanation: In the above code, the value of `x` is 23.456 and we are printing the value of `x` using the `%.2f` format specifier. `%.2f` rounds the value and prints the 2 digits after the decimal point.

39. What will be the output of the following C code?

```
#include <stdio.h>
void main()
{
    int x = 10;
    int y = x++ + 20;

    printf("%d,%d",x,y);

    return 0;
}
```

A. 11,30
B. 11,31
C. 10,30
D. 10,31

ANSWER:- A) 11,30

Explanation: In the above code, we are using a post-increment statement (`x++`), post-increment increases the value after evaluating the current expression. Thus, the value of `y` will be 30 and then `x` will be 11.

40. Increment (++) and decrement (--) are the ___ operators in C?

A. Unary
B. Binary
C. Ternary
D. None of the above

ANSWER:- A) Unary

Explanation: Increment (++) and decrement (--) are the unary operators. They need one operand to perform the operation.

41. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    unsigned char c=290;
    printf("%d",c);
    return 0;
}
```

A. 290
B. 256
C. 34
D. Garbage

ANSWER:- C) 34

Explanation:290 is beyond the range of unsigned char. Its corresponding value printed is: (290 % (UCHAR_MAX +1) where UCHAR_MAX represents highest (maximum) value of unsigned char type of variable. The value of UCHAR_MAX=255. Thus it prints 290 % (UCHAR_MAX+1)=34

42. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a=0;

    a=5||2|1;
    printf("%d",a);

    return 0;
}
```

A. 1
B. 7
C. 0
D. 8

ANSWER:- A) 1

Explanation: Bitwise OR operator (|) has precedence over logical OR operator (||). Thus the expression 5 || 2 | 1 is actually 5 || (2 |1).

Now,

```
2= 0000 0010
1= 0000 0001
2|1= 0000 0011=3
5 || 3 returns true as both are
nonzero
Thus a=1
```

43. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int x =-100;

    -100;
    printf("%d",x);

    return 0;
}
```

A. 100
B. -100
C. 0
D. Error

ANSWER:- B) -100

Explanation: The statement "-100;" is evaluated and this does not affect the value of "x".

44. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int a,b,c;

    a=0x10; b=010;
    c=a+b;

    printf("%d",c);

    return 0;
}
```

A. 20
B. 24
C. Garbage
D. Error

ANSWER:- B) 24

Explanation:0x10 is hex value it's decimal value is 16 and 010 is an octal value it's decimal value is 8, hence answer will be 24.

45. Which C keyword is used to extend the visibility of variables?

A. extend
B. extends
C. extern
D. auto

ANSWER:- C) extern

Explanation: The "extern: keyword used to define an extern variable, that can be accessed in any source file. i.e., extern is used to extend the visibility of variables in C language.

46. What is the name of "&" operator in C?

A. Ampersand
B. And
C. Address of
D. None of the above

ANSWER:- C) Address of

Explanation: The "**&**" operator is known as 'Address Of' operator which is used to access the address of a variable.

C Conditional Statements MCQs

47. Which of the following are valid decision-making statements in C?

A. if
B. switch
C. nested if
D. All of these

ANSWER:- D) All of these

Explanation: All valid decision-making statements in C program are:

- if statement
- if-else statement
- nested if statement
- switch statement
- nested switch statement

48. Decision making in the C programming language is ___.

A. Repeating the same statement multiple times
B. Executing a set of statements based on some condition
C. Providing a name of the block of code
D. All of these

ANSWER:- B) Executing a set of statements based on some condition

Explanation: Decision-making in C programming is executing a block of code to be executed by the program based on some condition.

49. Which of the following is a true value in C programming?

A. 1
B. "includehelp"
C. ! NULL
D. All of these

ANSWER:- D) All of these

Explanation: All non-zero and non-null values in C programming are true.

50. Ternary operator in C programming is ___.

A. if-else-if
B. ? :
C. ? ; ?
D. None of these

ANSWER:- B) ? :

Explanation: Ternary operator is used to execute expressions based on the given condition.

51. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    printf((43 > 43) ? "value 1 is greater!" : "value 1 is not greater!");
    return 0;
}
```

A. value 1 is not greater
B. value 1 is greater
C. Error
D. None of these

ANSWER:- A) value 1 is not greater

52. What is the correct syntax of if statement in C program?

A. if(condition){
}
B. if(condition) :
C. If { [condition] }
D. None of these

ANSWER:- A)

```
if(condition){
}
```

Explanation: The correct syntax of if statement in C program is:

```
if(condition){
    // code to be executed
}
```

53. The if statement is a conditional statement?

A. True
B. False

ANSWER:- A) True

Explanation: The if statement is a conditional statement, i.e., the block is executed based on the given condition.

54. When the condition of if statement is false, the flow of code will ___.

A. go into the if block
B. Exit the program
C. Continue the code after skipping the if block
D. None of these

ANSWER:- C) Continue the code after skipping the if block

Explanation: When the condition of if statement is false, the code after the if block will be executed and the if block code is skipped.

55. What will be the result of the following condition?

```
(! (25 > 25))
```

A. True
B. False
C. Error
D. None of these

ANSWER:- A) True

56. Which statement is required to execute a block of code when the condition is false?

A. for
B. if
C. else
D. All of these

ANSWER:- C) else

Explanation: In the if-else block, the if block is executed when condition is True and else block is executed when condition is false.

57. Can the else statement exist without the if statement in C?

A. Yes
B. No

ANSWER:- B) No

Explanation: The else statement needs an if statement to execute the block.

58. Which of these if...else block syntax is correct?

A.
```
if(condition){
}
else {
}
```
B.
```
if(condition){
}
else(condition){
}
```
C.
```
if{
}
else {
}
```
D. None of these

ANSWER:- A)

```
if(condition){
}
else {
}
```

Explanation: The syntax of if...else block of code in C is:

```
if(condition){
    // true block
}
else {
    // false block
}
```

59. The if-elseif-else statement in C programming is used?

A. Create multiple conditional statements
B. Return values
C. Loop in if-else block
D. All of these

ANSWER:- A) Create multiple conditional statements

Explanation: The if-elseif-else statement is a statement which contains multiple statements based on conditions.

60. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    int marks = 43;

    if (marks > 90)
        printf("Grade : A ");
    else if (marks > 75)
        printf("Grade : B ");
    else if (marks > 60)
        printf("Grade : C ");
    if (marks > 40)
        printf("Grade : D ");
    else
        printf("Fail ");
    return 0;
}
```

A. Grade : A
B. Grade : B
C. Grade : C
D. Grade : D

ANSWER:- D) Grade : D

61. How many expressions can be checked using if...elseif...else statement?

A. 100
B. 1
C. Infinite
D. None of these

ANSWER:- C) Infinite

Explanation: You can execute any number of expressions using if-elseif-else statements, each elseif statement containing one condition.

62. Is it possible to nest if-else statements in C programming?

A. Yes
B. No

ANSWER:- A) Yes

Explanation: Nesting of if-else statements in C programming is possible.

63. Which of the following syntax is correct for nested if-else statements?

```
A.  if(exp1){
B.    if(exp2){
C.    }
D.  }
E.  else {
F.    if(exp3){
G.    }
H.  }
I.  if(exp1){
J.  }else {
K.  }
```

L.

```
M.  if{}
```

N. None of these

ANSWER:- A)

```
if(exp1){
          if(exp2){
          }
}
else {
          if(exp3){
          }
}
```

Explanation: The correct syntax for nested if-else statements in C programming is:

```
if(exp1){
          if(exp2){
          }
}
else {
          if(exp3){
          }
}
```

64. What will be the output of the following C code?

```
#include <stdio.h>
int main(){
    int n = 65;
    if (n >= 75) {
        if (n >= 95) {
            printf("Excellent");
        }
        else
            printf("Pass");
    }
    else
        printf("Fail");
```

```
}
```

A. Excellent
B. Pass
C. Fail
D. None of these

ANSWER:- C) Fail

65. Multiple values of the same variable can be tested using ___.

A. switch
B. for
C. Function
D. All of these

ANSWER:- A) switch

Explanation: The switch statement in C is used to test for multiple values of a variable.

66. Without a break statement in switch what will happen?

A. All cases will work properly
B. Cases will fall through after matching the first check
C. Switch will throw error
D. All of these

ANSWER:- B) Cases will fall through after matching the first check

Explanation: The break statement is used to terminate the current flow of code. And if it is not present in the switch statement, the cases will execute all cases after the matched case.

67. When all cases are unmatched which case is matched in a switch statement?

A. Default case
B. First case
C. No case
D. None of these

ANSWER:- A) Default case

Explanation: The default case of switch statement is executed when no other case is matched.

68. What will be the output of the following C code?

```c
#include <stdio.h>
int main(){
    char grade = 'B';

    switch (grade) {
    case 'A':
        printf("Excellent!\n");
    case 'B':
    case 'C':
        printf("Well done\n");
    case 'D':
        printf("You passed\n");
    case 'F':
        printf("Better try again\n");
        break;
    default:
        printf("Invalid grade\n");
    }
}
```

A. Well done
B. You passed
C. Better try again
D. All of these

ANSWER:- D) All of these

Explanation: There is no break statement in case B, Case C, case D. the code will fall through executing all the print statements.

4) C Control Statements MCQs

69. Loops in C programming are used to ___.

A. Execute a statement based on a condition
B. Execute a block of code repeatedly
C. Create a variable
D. None of these

ANSWER:- B) Execute a block of code repeatedly

Explanation: Loops in programming are used to execute a block of code repeatedly.

70. Which of these is an exit-controlled loop?

A. for
B. if
C. do...while
D. while

ANSWER:- C) do...while

Explanation: The do...while loop check for a condition after executing the loop block once. Hence, it is called an exit-controlled loop.

71. Which statements are used to change the execution sequence?

A. Loop control statement
B. Function statement
C. Conditional statement
D. All of these

ANSWER:- A) Loop control statement

Explanation: Loop control statements in C are used to change the execution sequence of the loop.

72. What will happen if the loop condition will never become false?

A. Program will throw an error
B. Program will loop infinitely
C. Loop will not run
D. None of these

ANSWER:- B) Program will loop infinitely

Explanation: An infinite loop in a program is a condition when the loop continues when the loop continues to run infinitely because the condition never becomes false.

73. Which of these statements is correct in case of while loop in C?

A. Executes the block till the condition become false
B. Is an entry controlled loop
C. There might be condition when the loop will not execute at all
D. All of these

ANSWER:- D) All of these

Explanation: All of the above statements are true in the case of while loop. While loop is an entry-controlled loop, the block executes when the condition becomes false.

74. Which of the following is valid syntax for creating a while loop?

```
A. while{
B. } (condition)
```

C.

```
D. while(condition){
E. }
```

F.

```
G. while{
H. }
```

I.
J. All of these

ANSWER:- B)

```
while(condition){
}
```

Explanation: The correct syntax for creating a while loop is:

```
while(condition){
}
```

75. What will be the output of the following C code?

```
#include <stdio.h>
int main(){
    int a = 11;

    while (a < 20) {
        printf("%d  ", a);
        a += 2;
    }
    return 0;
}
```

A. 11 13 15 17 19
B. 11 12 13 14 15 16 17 18 19 20
C. 11 13 15 17 19 21
D. None of these

ANSWER:- A) 11 13 15 17 19

76. Which loop executes the block a specific number of times?

A. while loop
B. for loop
C. do...while loop
D. All of these

ANSWER:- B) for loop

Explanation: The for loop executes the block a specific number of times.

77. Which of the following parts of the for loop can be eliminated in C?

A. init

B. condition
C. increment
D. All of these

ANSWER:- D) All of these

Explanation: Syntax of for loop:

```
for(init, condition, increment){
}
```

Inside the initialization statement (init), any of the three init or condition or increment can be eliminated i.e., all are optional. The loop can work without them also.

78. When all parts of the for loop are eliminated, what will happen?

A. For loop will not work
B. Infinite for loop
C. Error
D. None of these

ANSWER:- B) Infinite for loop
Explanation: On eliminating all the parts of a for loop an infinite for loop will run.

79. When the condition of the do-while loop is false, how many times will it execute the code?

A. 0
B. 1
C. Infinite
D. All of these

ANSWER:- B) 1
Explanation: The do-while loop is an exit-controlled loop, hence it will run at least once, even if the condition becomes false.

80. Can a loop be nested in C programming?

A. Yes
B. No

ANSWER:- A) Yes

Explanation: Yes, the nesting of loop is possible in C programming language.

81. What will be the output of the following C code?

```c
#include <stdio.h>

int main(){
    int i, j;

    for (i = 2; i < 10; i++) {
        for (j = 2; j <= (i / j); j++)
            if (!(i % j))
                break;
        if (j > (i / j))
            printf("%d ", i);
    }

    return 0;
}
```

A. 2 3 4 5 6 7 8 9
B. 3 5 7 9
C. 2 3 5 7
D. 2 3 5 7 11

ANSWER:- C) 2 3 5 7
82. A string is terminated by ___.

A. Newline ('\n')
B. Null ('\0')
C. Whitespace
D. None of the above

ANSWER:- B) Null ('\0')

Explanation: In C programming language, a string is a sequence of characters terminated with a null character \0.

83. Consider the below statement, can we assign a string to variable like this:

```
char c[100];
c = "C programming";
```

A. Yes
B. No

ANSWER:- B) No

Explanation: No, we cannot assign a string like this. Because string is a character array and array type is not assignable. To assign a string to the character array, we need to use `strcpy()` function.

84. Which format specifier is used to read and print the string using printf() and scanf() in C?

A. %c
B. %str

C. %p
D. %s

ANSWER:- D) %s

Explanation: The format specifier "%s" is used to read and print the string using printf() and scanf() in C.

Example:

```
#include <stdio.h>

int main()
{
    char name[30];
    printf("Input name: ");
    scanf("%s", name);
    printf("Given name is: %s", name);
    return 0;
}

/*
Output:
Input name: Alvin
Given name is: Alvin
*/
```

85. Which function is used to read a line of text including spaces from the user in C?

A. scanf()
B. getc()
C. fgets()
D. All of the above

ANSWER:- C) fgets()

Explanation: In C programming language, the fgets() function can be used to read a line of text including spaces from the user.

Syntax:

```
fgets(variable_name, sizeof(size), stdin);
```

86. Which function is used to concatenate two strings in C?

A. concat()
B. cat()
C. stringcat()
D. strcat()

ANSWER:- D) strcat()

Explanation: In C programming language, the strcat() function is used to concatenate two strings.

Syntax:

```
strcat(s1, s2);
```

This function concatenates string s2 onto the end of string s1.

87. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char str1[] = { 'H', 'e', 'l', 'l',
'o' };
    char str2[] = "Hello";
    printf("%ld,%ld", sizeof(str1),
sizeof(str2));

    return 0;
}
```

A. 5,5
B. 6,6
C. 5,6
D. None of the above

ANSWER:- C) 5,6

Explanation:*str1* is initialized with the characters and there are only 5 characters. Thus, the length of the *str* is 5. While, *str2* is initialized with the string "Hello", when we initialized the string in this way - a null ('\0') character is inserted after the string. Thus, the length of *str2* is 6.

88. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
    char str1[] = "Hello";
    char str2[10];
    str2 = str1;
    printf("%s,%s", str1, str2);

    return 0;
}
```

A. Hello,
B. Hello,Hello
C. Hello,HelloHello

D. Error

ANSWER:- D) Error

Explanation: There will be a compilation error, because we cannot assign a string like this (str2 = str1). To resolve this issue, we have to use strcpy(str2,str1).

The output will be:

```
main.c:8:10: error: assignment to
expression with array type
    8 |      str2 = str1;
      |           ^
```

89. What will be the output of the following C code? (If the input is "Hello world")

```c
#include <stdio.h>
int main()
{
    char str[30];
    scanf("%s", str);
    printf("%s", str);
    return 0;
}
```

A. Hello world
B. Hello
C. Hello world\0
D. Error

ANSWER:- B) Hello

Explanation: When we read a string using the *scanf()* function, the input is terminated by the whitespace. Here, the input is "Hello world", so only "Hello" will be stored to *str*. Thus, the output will be "Hello".

90. Which function is used to compare two strings in C?

A. strcmp()
B. strcmpi()
C. compare()
D. cmpi()

ANSWER:- A) strcmp()

Explanation: The function *strcmp()* is used to compare two strings in C language.

The *strcmp()* is a built-in library function and is declared in *<string.h>* header file. This function takes two strings as arguments and compare these two strings lexicographically.

Syntax:

```c
int strcmp(const char* str1, const
char* str2);
```

91. Which function is used to compare two strings with ignoring case in C?

A. strcmp()
B. strcmpi()
C. compare()
D. cmpi()

ANSWER:- B) strcmpi()

Explanation: The function *strcmpi()* is used to compare two strings with ignoring case in C language.

The *strcmpi()* is a built-in library function and is declared in *<string.h>* header file. This function takes two strings as arguments and compare these two strings, it is similar to *strcmp()* function but the only difference is that *strcmpi()* function is not case sensitive.

Syntax:

```c
int strcmpi(const char* str1, const
char* str2);
```

6) C Arrays MCQs

92. Which is the correct syntax to declare an array in C?

A. data_type array_name[array_size];
B. data_type array_name{array_size};
C. data_type array_name[];
D. All of the above

ANSWER:- A) data_type array_name[array_size];

Explanation: The correct syntax to declare an array in C is:

```c
data_type array_name[array_size];
```

93. You can access elements of an array by ___.

A. values
B. indices
C. memory addresses
D. All of the above

ANSWER:- B) indices

Explanation: You can access elements of an array by indices. Array indices starts from 0 to array_size -1.

94. Which is/are the correct syntax to initialize an array in C?

A. data_type array_name[array_size] = {value1, value2, value3, ...};
B. data_type array_name[] = {value1, value2, value3, ...};
C. data_type array_name[array_size] = {};
D. Both A and B

ANSWER:- D) Both A and B

Explanation: Both the options A and B are correct to initialize an array.

Example:

```
int mark[5] = {19, 10, 8, 17, 9};
// or
int mark[] = {19, 10, 8, 17, 9};
```

95. Array elements are always stored in ___ memory locations.

A. Random
B. Sequential
C. Both A and B
D. None of the above

ANSWER:- B) Sequential

Explanation: Array elements are always stored in sequential memory locations.

96. Let x is an integer array with three elements having value 10, 20, and 30. What will be the output of the following statement?

```
printf("%u",x);
```

A. Prints the value of 0^{th} element (i.e., 10)
B. Prints the garbage value
C. An error occurs
D. Print the address of the array (i.e., the address of first (0^{th}) element

ANSWER:- D) Print the address of the array (i.e., the address of first (0^{th}) element

Explanation: If we print the array (in this case, x). The output will be the memory address of the array (i.e., the address of first (0^{th}) element.

97. What will be the output of the following C program?

```
#include <stdio.h>

int main()
{
    int x[5] = { 10, 20, 30 };
    printf("%d", x[3]);
    return 0;
}
```

A. 0
B. 30
C. Garbage value
D. Error

ANSWER:- A) 0

Explanation: In C language, when an array is partially initialized at the time of declaration then the remaining elements of the array is initialized to 0 by default.

98. What will be the output of the following C program?

```
#include <stdio.h>
int main()
{
    int x[5] = { 10, 20, 30 };
    printf("%d", x[-1]);
    return 0;
}
```

A. 0
B. 10
C. Garbage value
D. Error

ANSWER:- C) Garbage value

Explanation:C language compiler does not check array with its bounds, when an index is out of the range the garbage value is printed.

99. What will be the output of the following C program?

```
#include <stdio.h>
int main()
{
    int x[5] = { 10, 20, 30 };
    printf("%ld",
sizeof(x)/sizeof(x[0]));
    return 0;
}
```

A. 3
B. 4
C. 5
D. 6

ANSWER:- C) 5

Explanation: The statement sizeof(x)/sizeof(x[0]) can be used to get the total numbers of elements, sizeof(x) will return the size of array while sizeof(x[0]) will return the size of first element i.e., size of the type. Their division will return the total number of elements.

100. What will be the output of the following C program?

```
#include <stdio.h>
int main()
{
    int x[5] = { 10, 20, 30 };
    printf("%d", x[3]);
    return 0;
}
```

A. 0
B. 30
C. Garbage value
D. Error

ANSWER:- A) 0

Explanation: In C language, when an array is partially initialized at the time of declaration then the remaining elements of the array is initialized to 0 by default.

101. What will be the output of the following C program?

```
#include <stdio.h>
int main()
{
    int x[5] = { 10, 20, 30 };
    printf("%d", x[-1]);
    return 0;
}
```

A. 0
B. 10
C. Garbage value
D. Error

ANSWER:- C) Garbage value

Explanation:C language compiler does not check array with its bounds, when an index is out of the range the garbage value is printed.

102. If we pass an array as an argument to a function, what actually gets passed?

A. Value of elements in array
B. First element of the array
C. Base address of the array i.e., the address of the first element
D. Address of the last element of array

ANSWER:- C) Base address of the array i.e., the address of the first element

Explanation: If we pass an array as an argument to a function – the base address of the array (the address of the first element) is passed to the function.

7) C Structures and Union MCQs

103. Which of the following is the collection of different data types?

A. structure
B. string
C. array
D. All of the above

ANSWER:- A) structure

Explanation: In C programming language, a structure is a user-defined data type which is the collection of different data types.

104. Which operator is used to access the member of a structure?

A. -
B. >
C. *
D. .

ANSWER:- D) .

Explanation: The dot (.) operator is used to access the member of a structure.

105. Which of these is a user-defined data type in C?

A. int
B. union
C. char
D. All of these

ANSWER:- B) union

Explanation: Union is a user defined data type in C.

106. "A union can contain data of different data types". True or False?

A. True
B. False

ANSWER:- A) True

Explanation: Union is a user defined data structure which can store data variables of different data types.

107. Which keyword is used to define a union?

A. un
B. union
C. Union
D. None of these

ANSWER:- B) union

Explanation: The union is defined using the keyword union.

108. The size of a union is ___.

A. Sum of sizes of all members
B. Predefined by the compiler
C. Equal to size of largest data type
D. None of these

ANSWER:- C) Equal to size of largest data type

Explanation: The size of the union is equal to the size of the largest data type declared in the union.

109. All members of union ___.

A. Stored in consecutive memory location
B. Share same memory location
C. Store at different location
D. All of these

ANSWER:- B) Share same memory location

Explanation: All members of a union in C are stored at the same memory location.

110. Which of the below statements is incorrect in case of union?

A. Union is a user-defined data structure
B. All data share same memory
C. Union stores methods too
D. union keyword is used to initialize

ANSWER:- C) Union stores methods too

Explanation: Unions in C is a data structure which store data types but not the methods.

111. The members of union can be accessed using ___.

A. Dot Operator (.)
B. And Operator (&)
C. Asterisk Operator (*)
D. Right Shift Operator (>)

ANSWER:- A) Dot Operator (.)

Explanation: The dot operator (.) is used to access members of a union.

112. In which case union is better than structure?

A. Less memory is available
B. Faster compilation is required
C. When functions are included
D. None of these

ANSWER:- A) Less memory is available

113. Which is the correct syntax to create a union?

A. union union_name {
   };
B. union union_name {
   }
C. union union_name (
   );

D. Union union_name ( )

ANSWER:- A)

```
union union_name {
};
```

Explanation: The correct syntax of creating a union is:

```
union union_name {
};
```

114. What will be the output of the following C program?

```
#include <stdio.h>
union values {
    int val1;
    char val2;
} myVal;
int main()
{
    myVal.val1 = 66;

    printf("val1 = %p", &myVal.val1);
    printf("\nval2 = %p", &myVal.val2);
    return 0;
}
```

A. val1 = 0x54ac88dd2012
   val2 = 0x55ac76dd2014
B. Error
C. val1 = 0x55ac76dd2014
   val2 = 0x55ac76dd2014
D. Exception

ANSWER:- C) val1 = 0x55ac76dd2014
val2 = 0x55ac76dd2014

8) C Functions MCQs

115. What is a function in C?

A. User defined data type
B. Block of code which can be reused
C. Declaration syntax
D. None of these

ANSWER:- B) Block of code which can be reused

Explanation: Function in C is a block of code which can be reused.

116. Functions in C can accept multiple parameters. True or False?

A. True
B. False

ANSWER:- A) True

Explanation: Functions in C programming can accept multiple parameters.

117. Which keyword is used to return values from function?

A. Return
B. Value
C. Return type
D. All of these

ANSWER:- C) Return type

Explanation: The return in declaration is initialized using return type.

118. Which of these is not a valid parameter passing method in C?

A. Call by value
B. Call by reference
C. Call by pointer
D. All of these

ANSWER:- C) Call by pointer

Explanation: Valid parameter passing method in C is:

- Call by value
- Call by reference

119. A C program contains ___.

A. At least one function
B. No function
C. No value from command line
D. All of these

ANSWER:- A) At least one function

Explanation: The main() function is required in the C program.

120. A recursive function in C ___.

A. Call itself again and again

B. Loop over a parameter
C. Return multiple values
D. None of these

ANSWER:- A) Call itself again and again

Explanation: Recursive function in C is a function which calls itself again and again.

121. The scope of a function is limited to?

A. Current block
B. Only function
C. Whole file
D. Directory

ANSWER:- C) Whole file

Explanation: The scope of a function is limited to the file it is declared in.

122. Which of the below syntax is the correct way of declaring a function?

A. return function_name () {
}
B. data_type function_name (parameter) {
}
C. Void function_name (
)
D. None of these

ANSWER:- B)

```
data_type function_name (parameter){
}
```

Explanation:

The correct syntax of declaring a function in C:

```
data_type function_name (parameter){
}
```

123. The sqrt() function is used to calculate which value?

A. Square
B. Square of reverse bits
C. Square root
D. None of these

ANSWER:- C) Square root

Explanation: The sqrt() method in C is used to calculate the square root of a number.

124. What will be the output of the following C program?

```c
#include <stdio.h>
int myFunc(int x){
    return (--x);
}
int main(){
    int a = myFunc(13);
    printf("%d", a);
    return 0;
}
```

A. 13
B. 12
C. 14
D. Error

ANSWER:- B) 12

9) C Pointers MCQs

125. Before using a pointer variable, it should be ___.

A. Declared
B. Initialized
C. Both A. and B.
D. None of the above

ANSWER:- C) Both A. and B.

Explanation:A pointer variable should be declared and initialized before the using.

126. An uninitialized pointer in C is called ___.

A. Void pointer
B. Empty pointer
C. Invalid pointer
D. Wild pointer

ANSWER:- D) Wild pointer

Explanation: An uninitialized pointer in C is called wild pointer. Since, the pointer is initialized it may lead a program to behave wrongly or to crash.

127. ___ is a pointer that occurs at the time when the object is de-allocated from memory without modifying the value of the pointer.

A. Dangling pointer
B. Wild pointer
C. Void pointer
D. Null pointer

ANSWER:- A) Dangling pointer

Explanation:A dangling pointer is a pointer that occurs at the time when the object is de-allocated from memory without modifying the value of the pointer.

128. A ___ can be assigned the address of any data type.

A. Dangling pointer
B. Wild pointer
C. Void pointer
D. Null pointer

ANSWER:- C) Void pointer

Explanation:A void pointer can be assigned the address of any data type.

129. Which pointer is called general-purpose pointer?

A. Dangling pointer
B. Wild pointer
C. Void pointer
D. Null pointer

ANSWER:- C) Void pointer

Explanation:A void pointer is called general-purpose, because it can be used with any type of variable.

130. Pointer arithmetic is not possible on ___.

A. Integer pointers
B. Float pointers
C. Character pointers
D. Void pointers

ANSWER:- D) Void pointers

Explanation:

Pointer arithmetic is not possible on void pointers due to lack of concrete value and size.

131. Comment on the following pointer declaration?

```
int *p, x;
```

A. p is a pointer to integer, x is not
B. p and x, both are pointers to integer
C. p is pointer to integer, x may or may not be
D. p and x both are not pointers to integer

ANSWER:- A) p is a pointer to integer, x is not

Explanation: In the above given statement, variable `p` is a pointer to integer, while `x` is an integer variable.

132. What will be the output of the following C code?

```
#include <stdio.h>
int main(){
    int x = 10, *ptr;
    ptr = &x;
    *ptr = 20;
    printf("%d", x);
}
```

A. 10
B. 20
C. Error
D. A garbage value

ANSWER:- B) 20

Explanation: In the above program, `x` contains 10 and `ptr` is a pointer to `x`. We are changing the value of `x` with the help of pointer. Thus, the value will be changed and it will be 20.

133. What will be the output of the following C code?

```
#include <stdio.h>
int main(){
    char* ptr;
    ptr = "IncludeHelp";
    printf("%c", *&*ptr);
    return 0;
}
```

A. IncludeHelp
B. I
C. Some address
D. Error

ANSWER:- B) I

Explanation: In the above program, in printf statement – we are using two symbols those are * and &. The * is a dereference operator, and the & is a reference operator. These symbols can be used any number of times. Here ptr points to the first character in the string "IncludeHelp". *ptr dereferences it and so its value is I. Again & references it to an address and * dereferences it to the value I.

134. What is the correct syntax to declare pointer to pointer i.e., double pointer?

A. type **pointer_name;
B. type *&pointer_name;
C. type *(*pointer_name);
D. type **(pointer_name);

ANSWER:- A) type **pointer_name;

Explanation: The correct syntax syntax to declare pointer to pointer i.e., double pointer is:

```
type **pointer_name;
```

Example:

```
int **ptr; //declaration double
pointer
```

135. What is ptr in the given statement?

```
int (*ptr)[5];
```

A. ptr is an array of 5 pointers
B. ptr is a simple integer array
C. ptr is a pointer to a 5 elements integer array
D. None of the above

ANSWER:- C) ptr is a pointer to a 5 elements integer array

Explanation:

In the above statement, `ptr` is a pointer to a 5 elements integer array.

136. What is the correct syntax to declare a pointer to a constant?

A. const type *pointer_name;
B. type const *pointer_name;
C. Both A. and B.
D. None of the above

ANSWER:- A) const type *pointer_name;

Explanation: The correct syntax to declare a pointer to a constant is:

```
const type *pointer_name;
```

10) C File Handling MCQs

137. Which function is used to open a file in C?

A. open()
B. fopen()
C. file_open()
D. fileopen()

ANSWER:- B) fopen()

Explanation: In C programming language, the fopen() function is used to open a file.

Syntax:

```
ptr = fopen("fileopen","mode");
```

Where, `fileopen` is the name of the file, and `mode` is the mode of the file.

138. Which character(s) is/are used to open a binary file in append mode in C?

A. a
B. b
C. ba
D. ab

ANSWER:- D) ab

Explanation: The characters "ab" opens a binary file in append mode.

Syntax:

```
ptr = fopen("fileopen","ab");
```

139. Which character(s) is/are used to open a binary file in reading and writing mode in C?

A. rw
B. rwb

C. rb+
D. rwb

ANSWER:- C) rb+

Explanation: The characters "rb+" opens a binary file in reading and writing mode.

Syntax:

```
ptr = fopen("fileopen","rb+");
```

140. Which is the correct syntax to declare a file pointer in C?

A. File *file_pointer;
B. FILE *file_pointer;
C. File file_pointer;
D. FILE *file_pointer;

ANSWER:- B) FILE *file_pointer;

Explanation:

The correct syntax to declare a file pointer in C is:

```
FILE *file_pointer;
```

141. Which function is used to close an opened file in C?

A. close()
B. fclose()
C. file_close()
D. fileclose()

ANSWER:- B) fclose()

Explanation: In C programming language, the fclose() function is used to close an opened file.

Syntax:

```
fclose(file_pointer);
```

142. Function fwrite() works with ___.

A. Text files
B. Binary files
C. Both A. and B.
D. None of the above

ANSWER:- B) Binary files

Explanation: Function fwrite() works with binary files. The file should be opened/created in binary mode.

143. What is the value of EOF in C?

A. -1
B. 0
C. 1
D. Null

ANSWER:- A) -1

Explanation: The EOF stands for End of File which is a Macro defined in stdio.h header file. With the GNU C Library, EOF is -1. In other libraries, its value may be some other negative number.

144. Which function checks the end-of-file indicator for the given stream in C?

A. eof()
B. EOF
C. feof()
D. None of the above

ANSWER:- C) feof()

Explanation: The C library function feof() checks the end-of-file indicator for the given stream.

Syntax:

```
int feof(FILE *stream);
```

145. Which function is used to seek the file pointer position in C?

A. seek()
B. fseek()
C. fileseek()
D. fmove()

ANSWER:- B) fseek()

Explanation: In C programming language, the fseek() function is used to seek the file pointer position.

Syntax:

```
int fseek(FILE *file_pointer, long int
offset, int whence);
```

Where,

- file_pointer– This is the pointer to a FILE object that identifies the stream.
- offset – This is the number of bytes to offset from whence.
- whence – This is the position from where offset is added, the value of this parameter may SEEK_SET, SEEK_CUR, or SEEK_END.

146. Which function is used to delete an existing file in C?

A. delete()
B. fremove()
C. frem()
D. remove()

ANSWER:- D) remove()

Explanation: In C programming language, the remove() function is used to delete an existing file.

Syntax:

```
int remove(const char *filename);
```

Where, `filename` is the name of the file to be removed.

11) C Preprocessor MCQs

147. What is CPP in C programming?

A. C processing platform
B. C PreProcessor
C. C pre-platform
D. None of these

ANSWER:- B) C PreProcessor

Explanation: In C programming, C preprocessors are referred to as CPP.

148. Which symbol is used to begin a preprocessor?

A. @
B. !
C. #
D. All of these

ANSWER:- C) #

Explanation: All preprocessor commands begin with a hash symbol (#).

149. Which of these is a valid preprocessor in C?

A. #include
B. #if
C. #error
D. All of these

ANSWER:- D) All of these

150. (\) operator in C is ___.

A. Macro continuation operator
B. Stringize operator
C. Tokenizer
D. None of these

ANSWER:- A) Macro continuation operator

Explanation: Macro continuation (/) operator is used to continue a macro which is too long for a single line.

151. Which operator is used to stringize variables in C?

A. /
B. &
C. *
D. #

ANSWER:- D) #

Explanation:# Operator is used to stringize variables in C.

152. What will be the output of the following C code?

```
#include <stdio.h>
#define message_for(a, b) \
    printf("Learn " #a " programming
language at " #b)
int main(void){
    message_for(python, includehelp);
    return 0;
}
```

A. Learn python programming language at includehelp
B. Learn #a programming language at #b
C. Learn programming language at
D. None of these

ANSWER:- A) Learn python programming language at includehelp

153. Is it possible in a C program to merge two tokens into one token?

A. Yes
B. No

ANSWER:- A) Yes

Explanation: The ### preprocessor in C is used to merge two tokens to one.

154. Which of these is not a valid preprocessor in C?

A. \
B. ?
C. ###
D. None of these

ANSWER:- B) ?

Explanation:\ and ### are both valid preprocessors in C.

155. Header files ___.

A. Contain function declarations
B. Can be included to a program
C. End with .h extension
D. All of these

ANSWER:- D) All of these

Explanation: All statements are correct. The header files are special files included in a program that contain functions and macros which can be imported and used in the code. The extension of header files is ".h".

156. Can programmers create their own header files?

A. Yes
B. No

ANSWER:- A) Yes

Explanation:A header file can be created by a programmer with some function that she needs to use in other files as well.

157. Which of these is valid syntax to include a header in C?

A. #include <header>
B. #include "header"
C. Both A and B
D. All of these

ANSWER:- C) Both A and B

Explanation: The valid syntaxes to include a header file to a program in C are:

- #include <header>
- #include "header"

158. What will happen if a header file is included in a program twice?

A. Program will throw an error
B. Program will throw an exception
C. Program will run normally
D. None of these

ANSWER:- A) Program will throw an error

Explanation: When the same header is included twice in the program, the compiler will throw an error as its teens to process it twice.

159. Which syntax is correct to include a specific preprocessor based on configuration?

A. #defif system1 <system.h>
B. #include system1 "system.h"
C. import <system.h> if system1
D. All of these

ANSWER:- B) #include system1 "system.h"

Explanation:

The correct way to include a specific preprocessor based on configuration is known as computed include is:

```
#include system1 "system.h"
```

// Remening website to download MCQ

1. https://www.examveda.com/c-program/practice-mcq-question-on-c-fundamentals/
2. https://www.examtray.com/subject/c-language-mcq-questions-and-answers-topics

3. https://www.indiabix.com/c-programming/questions-and-answers/
4. https://www.geeksforgeeks.org/50-c-language-mcqs-with-answers/
5. https://www.interviewbit.com/c-programming-mcq/
6. https://www.javatpoint.com/c-language-mcq
7.