C Programming Part 1

MCQ=70

1. Who is father of C Language?
A. BjarneStroustrup
B. James A. Gosling
C. Dennis Ritchie
D. Dr. E.F. Codd
Answer: Option C

2. C Language developed at _____?

A. AT & T's Bell Laboratories of USA in 1972
B. AT & T's Bell Laboratories of USA in 1970
C. Sun Microsystems in 1973
D. Cambridge University in 1972
Answer: Option A

3. For 16-bit compiler allowable range for integer constants is _____?
A. -3.4e38 to 3.4e38
B. -32767 to 32768
C. -32668 to 32667
D. -32768 to 32767
Answer: Option D
Solution: In a 16 Bit C compiler we have 2 bytes to store an integer, and 1 byte for a character. For unsigned integers the range is 0 to 65535. **For signed integers the range is -32768 to 32767.** For unsigned character, 0 to 255

4. C programs are converted into machine language with the help of

A. An Editor
B. A compiler
C. An operating system
D. None of these.
Answer: Option B
Solution:A Compiler is a system software that converts high level language into machine level language.

5. C was primarily developed as

A. System programming language
B. General purpose language
C. Data processing language
D. None of the above.
Answer: Option A

Solution: C language was primarily developed write code for Unix Operating System. So Option A is correct answer

6. Standard ANSI C recognizes _____ number of keywords?

A. 30
B. 32
C. 24
D. 36
E. 40
Answer: Option B
Solution: Standard C has 32 keywords. Those keyword express different identical work. All keywords are store at library of C program.

7. Which one of the following is not a reserved keyword for C?

A. auto
B. case
C. main
D. default
E. register
Answer: Option C
Solution: * auto:- default storage class and variables declares inside the function.
case:- must be unique
register :- stored in the CPU register of the processor
default:- no associated value , only reserved value
main:- c program execution begins and main() is a Function. So it is not a reserved keyword.

8. A C variable cannot start with

A. A number
B. A special symbol other than underscore
C. Both of the above
D. An alphabet
Answer: Option C
Solution:

9. If integer needs two bytes of storage, then maximum value of an unsigned integer is
A. $2^{16} - 1$
B. $2^{15} - 1$
C. 2^{16}
D. 2^{15}
E. None of these
Answer: Option A
Solution: An integer is a number with no fractional part; it can be positive, negative or zero. In ordinary usage, one uses a minus sign to designate a negative integer. However, a computer can only store information in bits, which can only have the values zero or one. We might expect, therefore, that the storage of negative integers in a computer

might require some special technique. As you might imagine, an unsigned integer is either positive or zero. Consider a single digit decimal number: In a single decimal digit, you can write a number between 0 and 9. In two decimal digits, you can write a number between 0 and 99, and so on. Since nine is equivalent to 10^1 - 1, 99 is equivalent to 10^2 - 1, etc. In n decimal digits, you can write a number between 0 and 10^n - 1. So, analogously, in the binary number system, An unsigned integer containing n bits can have a value between 0 and 2^n - 1 (which is 2^n different values)

10. What is the correct value to return to the operating system upon the successful completion of a program?
A. 1
B. -1
C. 0
D. Program do no return a value.
E. 2
Answer: Option C
Solution: C Programming return 0. In C and C++ programs the main function is of type “int” and therefore it should return an integer value. The return value of the main function is considered the "Exit Status" of the application. On most operating systems returning 0 is a success status like saying "The program worked fine"

11. Which is the only function all C programs must contain?
A. start()
B. system()
C. main()
D. printf()
E. getch()
Answer: Option C
Solution:

12. Which of the following is not a correct variable type?
A. float
B. real
C. int
D. double
E. char
Answer: Option B
Solution:

13. What number would be shown on the screen after the following statements of C are executed?
```
charch;
int i;
ch = 'G';
i = ch-'A';
printf("%d", i);
```

A. 5

B. 6
C. 7
D. 8
E. 9
Answer: Option B
Solution: Since the ASCII value of G is 71. If A is 65 hence the difference of 71 and 65 is 6.

14. Find the output of the following program. void main() { int i=01289; printf("%d", i); }

A. 0289
B. 1289
C. 713
D. 0713
E. Syntax error
Answer: Option E
Solution: The prefix 0 in an integer value indicates octal value. In octal value use of 8 and 9 is not allowed and hence the error.

15. Find the output of the following program.

```
void main()
{
int i=065, j=65;
printf("%d %d", i, j);
}
```

A. 53 65
B. 65 65
C. 065 65
D. 053 65
E. Syntax error
Answer: Option A
Solution: As octal 65 (065) is equivalent of decimal value 53. So answer is 53 65.

16 What is the difference between a declaration and a definition of a variable?
A. Both can occur multiple times, but a declaration must occur first.
B. A definition occurs once, but a declaration may occur many times.
C. Both can occur multiple times, but a definition must occur first.
D. A declaration occurs once, but a definition may occur many times.
E. There is no difference between them.
Answer: Option B
Solution: Declaration of variable mean to tell compiler there is a varfuncstruct of particular data type. Definition of a variable mean asking compiler to allocate memory to variable or define storage for that variable. So you can define a variable only one time but you can declare it as many time you want.

17. Which of the following operator takes only integer operands?
A. +

B. *
C. /
D. %
E. None of these
Answer: Option D
Solution: The modulo operator % computes the remainder. When a=9 is divided by b=4, the remainder is 1. The % operator can only be used with integers.

18. Determine output:

```
void main()
{
int i=0, j=1, k=2, m;
    m = i++ || j++ || k++;
printf("%d %d %d %d", m, i, j, k);
}
```

A. 1 1 2 3
B. 1 1 2 2
C. 0 1 2 2
D. 0 1 2 3
E. None of these
Answer: Option B
Solution: In an expression involving || operator, evaluation takes place from **left to right** and will be stopped if one of its components evaluates to true(a non zero value). So in the given expression m = i++ || j++ || k++. It will be stop at j and assign the current value of j in m. therefore m = 1 , i = 1, j = 2 and k = 2 **(since k++ will not encounter. so its value remain 2)**

19. Determine output:

```
void main()
{
int c = - -2;
printf("c=%d", c);
}
```

A. 1
B. -2
C. 2
D. Error
Answer: Option C
Solution: Here unary minus (or negation) operator is used twice. Same math rules applies, ie. minus * minus = plus. Note: However you cannot give like --2. Because -- operator can only be applied to variables as a decrement operator (eg., i--). 2 is a constant and not a variable.

20. **Determine output:**

```
void main()
```

```
{ int i=10;
    i = !i>14;
printf("i=%d", i);
}
```

A. 10
B. 14
C. 0
D. 1
E. None of these

Answer: Option C

Solution: In the expression !i>14 , **NOT (!)** operator has more precedence than ' **>**' symbol. **!** is a unary logical operator. **!i** (!10) is 0 (not of true is false). 0>14 is false (zero).

21. **In C programming language, which of the following type of operators have the highest precedence**

A. Relational operators
B. Equality operators
C. Logical operators
D. Arithmetic operators

Answer: Option D

Solution: the sequence of priority is "arithmetic operators > relational operators > equality > logical operators"

22. **What will be the output of the following program?**

```
void main()

{
int a, b, c, d;
    a = 3;
    b = 5;
    c = a, b;
    d = (a, b);
printf("c=%d d=%d", c, d);
}
```

A. c=3 d=3
B. c=3 d=5
C. c=5 d=3
D. c=5 d=5

Answer: Option B

Solution: In the first statement, value of **a** will be 3, because **assignment operator (=) has more priority more than comma (,)**, thus 3 will be assigned to the variable **c**.

In the second statement, value of **d** will be 5, **because (3,5) are enclosed in braces, and braces has more priority than assignment (=) operator**. When multiple values are given with comma operator within the braces, then right most value is considered as result of the expression. Thus, 5 will be assigned to the variable **d**.

23. **Which of the following comments about the ++ operator are correct?**

A. It is a unary operator
B. The operand can come before or after the operator
C. It cannot be applied to an expression
D. It associates from the right
E. All of the above
Answer: Option E
Solution: In C, ++ and -- operators are called increment and decrement operators. They are unary operators needing only one operand. Hence ++ as well as -- operator can appear before or after the operand with same effect. That means both i++ and ++i will be equivalent.

24. **What will be the output of this program on an implementation where int occupies 2 bytes?**

```
#include <stdio.h>
void main()
{
int i = 3;
int j;
    j = sizeof(++i + ++i);
printf("i=%d j=%d", i, j);
}
```

A. i=4 j=2
B. i=3 j=2
C. i=5 j=2
D. the behavior is undefined
Answer: Option B
Solution: Evaluating ++i + ++i would produce undefined behavior, but the operand of sizeof is not evaluated, so i remains 3 throughout the program. The type of the expression (int) is reduced at compile time, and the size of this type (2) is assigned to j.

25.**What will be the output?**

```
void main(){ int a=10, b=20;
char x=1, y=0;
if(a,b,x,y){
printf("EXAM");    }
}
```

A. XAM is printed
B. exam is printed
C. Compiler Error

D. Nothing is printed

Answer: Option D

Solution: Whenever the operands are separated by comma and within parenthesis the rightmost value is considered. In this case y is the rightmost operand and 0 is its value therefore if(a,b,x,y)-->if y-->if 0-->false hence nothing gets printed.

26. Which operator has the lowest priority?

A. ++

B. %

C. +

D. ||

E. &&

Answer: Option D

Solution:

examveda.com

C Operator Precedence Table

This page lists C operators in order of *precedence* (highest to lowest). Their *associativity* indicates operators of equal precedence in an expression are applied.

Operator	Description	Assoc
( ) [ ] . -> ++ --	Parentheses (function call) (see Note 1) Brackets (array subscript) Member selection via object name Member selection via pointer Postfix increment/decrement (see Note 2)	left-t
++ -- + - ! ~ (*type*) * & sizeof	Prefix increment/decrement Unary plus/minus Logical negation/bitwise complement Cast (convert value to temporary value of *type*) Dereference Address (of operand) Determine size in bytes on this implementation	right-
* / %	Multiplication/division/modulus	left-t
+ -	Addition/subtraction	left-t
<< >>	Bitwise shift left, Bitwise shift right	left-t
< <= > >=	Relational less than/less than or equal to Relational greater than/greater than or equal to	left-t
== !=	Relational is equal to/is not equal to	left-t
&	Bitwise AND	left-t
^	Bitwise exdusive OR	left-t
|	Bitwise inclusive OR	left-t
&&	Logical AND	left-t
| |	Logical OR	left-t
? :	Temary conditional	right-
= += -= *= /= %= &= ^= |= <<= >>=	Assignment Addition/subtraction assignment Multiplication/division assignment Modulus/bitwise AND assignment Bitwise exdusive/indusive OR assignment Bitwise shift left/right assignment	right-
,	Comma (separate expressions)	left-t

Note 1:
Parentheses are also used to group sub-expressions to force a different precedence; such parenthetical expressions can be nested and are evaluate

27.What number will z in the sample code given below?

int z, x=5, y= -10, a=4, b=2;

z = x++ - --y*b/a;

A. 5

B. 6

C. 9

D. 10

E. 11

Answer: Option D
Solution: According to precedence table execution of the given operators are as follows:
1. x++(Postfix operator) i.e x will become 5
2. y--(Prefix operator) i.e y will become -11
3. * and / have same priority so they will be executed according to their associativity i.e left to right. So, *(Multiplication) will execute first and then /(division).
4. -(Subtraction)

So the complete expression would be
5 - (-11)*2/4 = 5 - (-22)/4 = 5 - (-5) = 5 + 5 = 10.

28. **What is the output of the following statements?**
int i = 0;
printf("%d %d", i, i++);
A. 0 1
B. 1 0
C. 0 0
D. 1 1
E. None of these
Answer: Option B
Solution: Since the evaluation is from **LIFO.** So when the print statement execute value of *i* = 0
Since its execute from Last in First Out when *i*++ will be execute first and print value 0 **(since its post increment)** and after printing 0 value of *i* become 1. So it its prints for 1 for next *i***. so 1 0.**

29. What is the output of the following statements?
int b=15, c=5, d=8, e=8, a;
a = b>c ? c>d ?12 : d>e ? 13 : 14 : 15;
printf("%d", a);
A. 13
B. 14
C. 15
D. 12
E. Garbage Value
Answer: Option B
Solution: (((**d>e ? 13 : 14) c>d ? 12 :14) b>c ? 14 : 15)=14**

30. What will be the output of the following code fragment?
void main()
{
printf("%x",-1<<4);
}
A. fff0
B. fff1
C. fff2

D. fff3
E. fff4
Answer: Option A
Solution: -1 will be represented in binary form as:
1111 1111 1111 1111
Now -1<<4 means 1 is Shifted towards left by 4 positions, hence it becomes:
1111 1111 1111 0000 in hexadecimal form - fff0.

31. **Find the output of the following program.**

```
#include
void main()
{
int y=10;
if(y++>9 && y++!=10 && y++>11)
printf("%d", y);
else
printf("%d", y);
}
```

A. 11
B. 12
C. 13
D. 14
E. Compilation error
Answer: Option C
Solution: All the three condition in ***if*** is true.
if(y++>9 && y++!=10 && y++>11) such as
1st condition : 10++ > 9 is true and value of y is increase by 1 **i.e y =11**
2nd condition : 11++ != 10 is also ture and value of y is increase by 1 **i.e y =12**
3rd condition : 12++ > 11 is also ture and value of y is increase by 1 **i.e y =13**
Therefore ***if*** is excuted and print the value of **y = 13**

32. **Find the output of the following program.**

```
#include<stdio.h>
void main()
{
int y=10;
if(y++>9 && y++!=11 && y++>11)
printf("%d", y);
else
```

```
printf("%d", y);
}
```
A. 11
B. 12
C. 13
D. 14
E. Compilation error
Answer: Option B
Solution: Since the second condition is false so, further conditions will not be checked, it will be skipped.
1st condition : 10++ > 9 is true and value of y is increase by 1 **i.e y =11**
2nd condition : 11++ != 11 is also false and value of y is increase by 1 **i.e y =12**
3rd condition : will not be checked
Therefore ***if*** is excuted and print the value of **y = 12**

33. **Determine output of the following program code.**

```
#include<stdio.h>
void main()
{
int a, b=7;
  a = b<4 ? b<<1 : ++b>4 ? 7>>1 : a;
printf("%d %d", a, b);
}
```
A. 3 7
B. 7 3
C. 8 3
D. 3 8
E. None of these
Answer: Option D
Solution: Right shift= **7>>1=3** So **(++b>4 ? 7>>1 : a;)= 3 and b=8. Then a =(b<4 ? b<<1 : 3)= 3 and b=8. So the answer is D= 3 8.**

34. Choose the correct output for the following program.

```
#include<stdio.h>
void main()
{
int a=10, b=11, c=13, d;
  d = (a=c, b+=a, c=a+b+c);
printf("%d %d %d %d", d, a, b, c);
}
```
A. 50, 13, 11, 13
B. 50, 13, 24, 50
C. 13, 10, 24, 50
D. 50, 13, 24, 13
E. 13, 13, 24, 13

Answer: Option B
Solution: For any comma separated expression the outcome is the right most part So d=50. Then LIFO.

35.
Consider the following program fragment, and choose the correct one

```
void main()
{
int a, b = 2, c;
  a = 2 * (b++);
  c = 2 * (++b);
}
```

A. a = 4, c = 8
B. a = 3, c = 8
C. b = 3, c = 6
D. a = 4, c = 6
E. b = 4, c = 6
Answer: Option A

36. **Which operator from the following has the lowest priority?**
A. Assignment operator
B. Division operator
C. Comma operator
D. Conditional operator
E. Unary-operator
Answer: Option C

37. **Identify the correct output of the following code:**

```
void main()
{
int w=10, x=5, y=3, z=3;
if( (w < x ) && (y=z++) )
printf("%d %d %d %d", w, x, y, z);
else
printf("%d %d %d %d", w, x, y, z);
}
```

A. 10 5 4 4
B. 10 5 3 3
C. 10 5 4 3
D. 10 5 3 4
E. 10 5 5 5
Answer: Option B
Solution: As the first condition (w< x) is false and the logical operator && is used, the entire relation becomes false and the second part (y = z++) is not executed. Then else will executed.

38. **Given b=110 and c=20, what is the value of 'a' after execution of the expression a=b-=c*=5?**

A. 450
B. 10
C. 110
D. -10
E. -110
Answer: Option B
Solution: As the expression is evaluated from right to left
a=b-=c*=5 = is the assignment operator and evaluates from right to left
c = c*5 =100
b = b-c =110-100 =10
a =b =10

39. **what will be the output when following code is executed?**
void main()

{
int a=10, b;
  b = a++ + ++a;
printf("%d %d %d %d", b, a++, a, ++a);
}
A. 12 10 11 13
B. 22 12 12 13
C. 22 11 11 11
D. 22 14 12 13
E. 22 13 14 14
Answer: Option E
Solution: If we apply LIFO then answer will be 22 13 13 13. But in code block the answer is 22 13 14 14.

40. **What is the output of given program if user enter value 99?**

```
#include<stdio.h>
void main()
{
        int i;
        printf("Enter a number:");
        scanf("%d", &i); // 99 is given as input.
        if(i%5 == 0){
                printf("nNumber entered is divisible by 5");
    }
}
```

A. Enter a number:99
B. Enter a number:99 Number is divisible by 5
C. complier error

D. Run time error
Answer: Option A
Solution: since this program isn't having any syntax error so program is executed. It is clearly seen that 99 is not divisible by 5. So if statement will not execute and program will terminate.

41. **What is the output of given program if user enter "xyz" ?**

```
#include
void main()
{
        float age, AgeInSeconds;
        printf("Enter your age:");
        scanf("%f", &age);
        AgeInSeconds = 365 * 24 * 60 * 60 * age;
        printf("You have lived for %f seconds", AgeInSeconds);
}
```

A. Enter your age: xyz You have lived for 0 seconds
B. Enter your age: xyz You have lived for 0.00000 seconds
C. Enter your age: xyz "after that program will stop"
D. Run time error
Answer: Option B
Solution: When we give scanf() a "%f" format string, that means "We want you to try and get us a floating point number. When we provide input like 'xyz', it's not going to match anything, because 'xyz' is not a valid floating-point number.

42
What is the output of given program if user enter "xyz" ?

```
#include
void main()
{
        float age, AgeInSeconds;
        int value;
        printf("Enter your age:");
        value=scanf("%f", &age);
        if(value==0){
                printf("\\nYour age is not valid");
        }
        AgeInSeconds = 365 * 24 * 60 * 60 * age;
        printf("\\n You have lived for %f seconds", AgeInSeconds);
}
```

A. Enter your age : xyz Your age is not valid
B. Enter your age: xyz You have lived for 0 seconds
C. Enter your age: xyz Your age is not valid
D. Complier error

Answer: Option C
Solution: When we give scanf() a "%f" format string, that means "We want you to try and get us a floating point number. When we provide input like 'xyz', it's not going to match anything, because 'xyz' is not a valid floating-point number.

43. **What will be the output of the given program?**

```
#include<stdio.h>
void main()
{
int  i=10;
printf("i=%d", i);
    {
int  i=20;
        printf("i=%d", i);
        i++;
        printf("i=%d", i);
    }
printf("i=%d", i);
}
```

A. 10 10 11 11
B. 10 20 21 21
C. 10 20 21 10
D. 10 20 21 20
Answer: Option C
Solution: The scope of second declaration of i is limited to the block in which it is defined. Outside of the block variable is not recognized.

44. **What will be the output given program?**

```
#include<stdio.h>
void main()
{
        int i = -10;
        for(;i;printf("%d ", i++));
}
```

A. -10 to -1
B. -10 to infinite
C. -10 to 0
D. Complier error
Answer: Option A
Solution: for loop can be initialized outside of the loop. Since until -1 value of i remain a non-zero value and hence the condition is true up to -1. But when i is further increases its value becomes 0 and condition

becomes false and loop stops there.
Note:In C any non-zero value(positive or negative) evaluates to **true** and only zero value is evaluates to **false**

45. **What will be the output of the given program?**

```
#include<stdio.h>
void main()
{
        int a=11,b=5;
        if(a=5) b++;
        printf("%d %d", ++a, b++);
}
```

A. 12 7
B. 5 6
C. 6 6
D. 6 7
E. 11 6
Answer: Option C
Solution: Here if condition evaluates to true as a non-zero value i.e 5 is assigned to a. So the value of a = 5 and after increment value of b = 6. In printf statement due to pre-increment of a value of a printed will be 6 and due to post-increment of b value of b printed will be 6 and not 7

46. **What will be the output of the given program?**

```
#include<stdio.h>
void main()
{
int value=0;
if(value)
printf("well done ");
printf("Anydesk");
}
```

A. well done Anydesk
B. Anydesk
C. complier error
D. None of these
Answer: Option B
Solution: As the value of variable **value** is zero so, it evaluates to false in the if condition.

47. **What will be the output of the given program?**

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

```
        int value1, value2=100, num=100;
        if(value1=value2%5) num=5;
        printf("%d %d %d", num, value1, value2);
}
```

A. 100 100 100
B. 5 0 20
C. 5 0 100
D. 100 0 100
E. 100 5 100
Answer: Option D
Solution(By Examveda Team)
Expression value2%5 is equal to 0 and this value assigned to value1.
Therefore if condition reduces to if(0) so it fails.
Therefore body of if will not be executed i.e num = 5 will not be assigned.
So at printfnum = 100 , value1 = 0 and value2 = 100.

48. **What will be the output of the given program?**

```
#include
void main()
{
        floatnum=5.6;
        switch(num){
                case 5:printf("5");
                case 6:printf("6");
                default :printf("0");
                        break;

        }
        printf("%d", num);
}
```

A. 5 5.600000
B. 6 5.600000
C. 0 5.600000
D. Complier error
Answer: Option D
Solution: Compiler error switch expression is not integral. switch statement cannot work on float value.
49. **What will be the output of given program?**

```
#include<stdio.h>
void main()
{
        int a=1;
        if("%d=hello", a);
}
```

A. complier error
B. no error no output
C. 0
D. 1
Answer: Option B
Solution The if is conditional and it checks for expression whether it is zero or non-zero so it doesn't print any data it is clear and it is clearly seen that there is not a single syntax error therefore no complier error. After compling the program will be executed but there is not a single printf statement so it is executed but will not give any output.

50. **What will be the output of given program?**

```
#include<stdio.h>
void main()
{
        int a=3;
        for(;a;printf("%d ", a--);
}
```

A. no output
B. 3 2 1 0
C. 3 2 1
D. infinity loop
Answer: Option C
Solution: Decrement operator a-- in for loop statement are executed until the condition is true.
So it is executed till "a" not equal to zero and printf statement inside for loop print a value of "a"

51. **What will be the output of the following piece of code?**

```
for(i = 0; i<10; i++);
printf("%d", i);
```

A. 10
B. 0123456789
C. Syntax error
D. 0
E. Infinite loop
Answer: Option A
Solution: Due to semicolon(;) at the end of the for loop, after 10 iterations for loop will be terminated and the result will be 10.

52. What will be the output of the following code?

```
#include
void main()
{
int s=0;
while(s++<10)
   {
```

if(s<4 && s<9)
continue;
printf("%dt", s);
}
}
A. 1 2 3 4 5 6 7 8 9
B. 1 2 3 10
C. 4 5 6 7 8 9 10
D. 4 5 6 7 8 9
E. None of these
Answer: Option C
Solution: **'If' statement only true when value of 's' is less then 4**

Value of S	While	If(value of s)	Print	Pri
0	0(true)	1(true)	Not executed	
1	1(true)	2(true)	Not executed	
2	2(true)	3(true)	Not executed	
3	3(true)	4(false)	Executed	
4	4(true)	5(false)	Executed	
5	5(true)	6(false)	Executed	
6	6(true)	7(false)	Executed	
7	7(true)	8(false)	Executed	
8	8(true)	9(false)	Executed	
9	9(true)	10(false)	Executed	

53. Which command is used to skip the rest of a loop and carry on from the top of the loop again?
A. break
B. resume
C. continue
D. skip
E. None of these
Answer: Option C
Solution: **Syntax: continue ;**
Causes control to pass to the end of the innermost enclosing while, do, or for statement, at that point the loop continuation condition is re-evaluated. Example:
for(i = 0; i < 20; i++){
if(array[i] == 0)
continue;
array[i] = 1/array[i];
}

54. What is the output of the following statements?
for(i=10; i++; i<15)
printf("%d ", i);
A. 10 11 12 13 14

B. 10 11 12 13 14 15
C. 9 10 11 12 13
D. Infinite loop
E. None of these
Answer: Option D
Solution: Since the condition is always true it will go to infinite loop.

55. The type of the controlling expression of a switch statement cannot be of the type .....
A. int
B. char
C. short
D. float
E. long
Answer: Option D
Solution: Rules for switch statement in C language
1) The *switch expression* must be of an integer or character type.
2) The *case value* must be an integer or character constant.
3) The *case value* can be used only inside the switch statement.
4) The *break statement* in switch case is not must. It is optional. If there is no break statement found in the case, all the cases will be executed present after the matched case. It is known as *fall through* the state of C switch statement.

56. What's wrong in the following statement, provided k is a variable of type int?
for(k = 2, k <=12, k++)

A. The increment should always be ++k .
B. The variable must always be the letter i when using a for loop.
C. There should be a semicolon at the end of the statement.
D. The variable k can't be initialized.
E. The commas should be semicolons.
Answer: Option E
Solution: In for loop the three statements parts are separated by two semicolons which are missing here.

57. **Find the output of the following program.**
```
#include<stdio.h>
void main()
{
int y=10;
if(y++>9 && y++!=10 && y++>11)
printf("%d", y);
else
printf("%d", y);
}
```
A. 11

B. 12
C. 13
D. 14
E. Compilation error
Answer: Option C
Solution:
Here if Statement is true and if statement executes left to right. Since its AND operator so all the condition should be check until it finds any false statement. Initially y = 10 In 1st condition: y++>9 which is true and y become 11 in next use. In 2nd condition: y++!=10 which is also true and y become 12 in next use In 3rd condition: y++>11 which is also true and y become 13 in next use After that printf statement will execute and print y = 13

58. **What will be the final value of the digit?**

```
void main()
{
int digit = 0;
for( ; digit <= 9; )
digit++;
digit  *= 2;
  --digit;
}
```

A. -1
B. 17
C. 19
D. 16
E. 20
Answer: Option C
Solution: First of all for loop have no braces so for loop on have only next line in its body.
for(; digit <= 9;)
digit++;
After completing for loop **digit = 10;**
next statement digit *= 2; i.e **digit = digit * 2 = 20**;
next statement digit--; i.e 20-- => 19
So final value of ***digit is 19***

59. **What will be the following code's output if choice = 'R'?**

```
switch(choice)
{
case 'R' : printf("RED");
case 'W' : printf("WHITE");
case 'B' : printf("BLUE");
default  :printf("ERROR");break;
```

```
}
```
A. RED
B. RED WHITE BLUE ERROR
C. RED ERROR
D. RED WHITE BLUE
E. ERROR
Answer: Option B
Solution:
As the first option is matching, the cases are evaluated till the break statement is encountered or end of switch statement is encountered.

60. **What will be printed if the following code is executed?**

```
void main()
{
int x=0;
for( ; ; )
  {
if( x++ == 4 ) break;
continue;
  }
printf("x=%d", x);
}
```

A. x=0
B. x=5
C. x=4
D. x=1
E. Error
Answer: Option B
Solution: When x is 4 then break the loop. After increment x++, x became 5. So the answer 5.

61. Array is a _____ data structure.
a. Non-linear
b. Primary
c. Linear
d. Data type
Answer: (c) Linear
Solution: An array is a non-primitive and linear data structure that only stores a similar data type.
62. Which of the following function is used to write the integer in a file?
a. getw()
b. putw()
c. int value
d. f_int()
Answer: (b) putw()
Solution: The putw() is used to write the integer in a file.

Syntax:
putw(int i, FILE *fp);

63. A global variable is declared _____.
A. Outside of the function
B. Inside of the function
C. With the function
D. Anywhere in the program
Answer: (A) Outside of the function
Solution: A global variable is a variable that is declared outside of the function. A global variable can be used in all functions.

64. Who defines the user-defined function?
A. Compiler
B. Computer
C. Compiler library
D. Users
Answer: (D) Users
Solution: The user-defined functions are those functions that are defined by the user while writing the program. The user can define these functions according to their needs.

65. Which of the following functions is already declared in the "header file"?
A. User-define function
B. Built-in function
C. C function
D. None of the these
Answer: (B) Built-in function
Solution: Built-in functions are those functions whose prototypes are preserved in the header file of the "C" programming language. These functions are called and executed only by typing their name in the program. For example, scanf(), printf(), strcat(), etc.

66. Which of the following operations cannot be performed in file handling?
A. Open the file
B. Read the file
C. To write a file
D. None of the these
Answer: (D) None of the these
Solution: File handling is a process in which data is stored in a file using a program. The following operations can be performed in file handling:

- Create a new file
- Open file
- Read the file
- Write the file
- Delete file

- File closing

Therefore, option (d) is the correct answer.

67. Which symbol is used to declare a pointer?
A. *
B. #
C. &
D. &&
Answer: (A) *
Solution: To declare a pointer variable, we use a '*' symbol before the pointer variable. For example
int *k;

68.8) Which of the following header files is used for character type function in C language?
A. <assert.h>
B. <ctype.h>
C. <iostream.h>
D. <locale.h>
Answer: (B) <ctype.h>
Solution: The <ctype.h> header file is used for character type function in C language.

69. **String concatenation means -**
A. Combining two strings.
B. Extracting a substring out of a string.
C. Partitioning the string into two strings.
D. Merging two strings.
E. Comparing the two strings to define the larger one.
Answer: Option A
Solution: Concatenation means combining two strings together (appending the second at the end of first one).

70. **The statement int **a;**

A. is illegal
B. is legal but meaningless
C. is syntactically and semantically correct
D. None of these.
Answer: Option C
Solution: a is pointer to a pointer to an integer.

Short Question:

1.Why is C called a mid-level programming language?

C is called a mid-level programming language because it binds the low level and high -level programming language. We can use C language as a System programming to develop the operating system as well as an Application programming to generate menu driven customer driven billing system

2. What is the use of printf() and scanf() functions?

printf(): The printf() function is used to print the integer, character, float and string values on to the screen.

Following are the format specifier:

- **%d**: It is a format specifier used to print an integer value.
- **%s**: It is a format specifier used to print a string.
- **%c**: It is a format specifier used to display a character value.
- **%f**: It is a format specifier used to display a floating point value.

scanf(): The scanf() function is used to take input from the user.

3.What is the difference between the local variable and global variable in C?

Following are the differences between a local variable and global variable:

Basis for comparison	Local variable	Global variable
Declaration	A variable which is declared inside function or block is known as a local variable.	A variable which is declared outside function or block is known as a global variable.
Scope	The scope of a variable is available within a function in which they are declared.	The scope of a variable is available throughout the program.
Access	Variables can be accessed only by those statements inside a function in which they are declared.	Any statement in the entire program can access variables.
Life	Life of a variable is created when the function block is entered and destroyed on its exit.	Life of a variable exists until the program is executing.

Storage	Variables are stored in a stack unless specified.	The compiler decides the storage location of a variable.

4. How do you construct an increment statement or decrement statement in C?

There are actually two ways you can do this. One is to use the increment operator ++ and decrement operator --. For example, the statement "x++" means to increment the value of x by 1. Likewise, the statement "x --" means to decrement the value of x by 1. Another way of writing increment statements is to use the conventional + plus sign or - minus sign. In the case of "x++", another way to write it is "x = x +1".

5. **What is a stack?**

A stack is one form of a data structure. Data is stored in stacks using the FILO (First In Last Out) approach. At any particular instance, only the top of the stack is accessible, which means that in order to retrieve data that is stored inside the stack, those on the upper part should be extracted first. Storing data in a stack is also referred to as a PUSH, while data retrieval is referred to as a POP.

6. **What is variable initialization and why is it important?**

This refers to the process wherein a variable is assigned an initial value before it is used in the program. Without initialization, a variable would have an unknown value, which can lead to unpredictable outputs when used in computations or other operations.

7. **Differentiate Source Codes from Object Codes**

Source codes are codes that were written by the programmer. It is made up of the commands and other English-like keywords that are supposed to instruct the computer what to do. However, computers would not be able to understand source codes. Therefore, source codes are compiled using a compiler. The resulting outputs are object codes, which are in a format that can be understood by the computer processor. In C programming, source codes are saved with the file extension .C, while object codes are saved with the file extension .OBJ

8. **What is the use of a '\0' character?**

It is referred to as a terminating null character, and is used primarily to show the end of a string value.

9. **What is the modulus operator?**

The modulus operator outputs the remainder of a division. It makes use of the percentage (%) symbol. For example: 10 % 3 = 1, meaning when you divide 10 by 3, the remainder is 1.

10. **What is a nested loop?**

A nested loop is a loop that runs within another loop. Put it in another sense, you have an inner loop that is inside an outer loop. In this scenario, the inner loop is performed a number of times as specified by the outer loop. For each turn on the outer loop, the inner loop is first performed.

11.**When is the "void" keyword used in a function?**

When declaring functions, you will decide whether that function would be returning a value or not. If that function will not return a value, such as when the purpose of a function is to display some outputs on the screen, then "void" is to be placed at the leftmost part of the function header. When a return value is expected after the function execution, the data type of the return value is placed instead of "void

12. What is the usage of the pointer in C?

- **Accessing array elements**: Pointers are used in traversing through an array of integers and strings. The string is an array of characters which is terminated by a null character '\0'.
- **Dynamic memory allocation**: Pointers are used in allocation and reallocation of memory during the execution of a program.
- **Call by Reference**: The pointers are used to pass a reference of a variable to other function.
- **Data Structures like a tree, graph, linked list, etc.**: The pointers are used to construct different data structures like tree, graph, linked list, etc.

13. What is a NULL pointer in C?

A pointer that doesn't refer to any address of value but NULL is known as a NULL pointer. When we assign a '0' value to a pointer of any type, then it becomes a Null pointer.

14. What is a far pointer in C?

A pointer which can access all the 16 segments (whole residence memory) of RAM is known as far pointer. A far pointer is a 32-bit pointer that obtains information outside the memory in a given section.

15.

What is a pointer on pointer?

It’s a pointer variable which can hold the address of another pointer variable. It de-refers twice to point to the data held by the designated pointer variable.Eg: int x = 5, *p=&x, **q=&p;

Therefore ‘x’ can be accessed by **q.

16. Distinguish between malloc() &calloc() memory allocation?

Both allocates memory from heap area/dynamic memory. By default calloc fills the allocated memory with 0’s.

17. What are the valid places for the keyword break to appear?

Break can appear only with in the looping control and switch statement. The purpose of the break is to bring the control out from the said blocks

18. How a negative integer is stored?

Get the two's compliment of the same positive integer. Eg: 1011 (-5)
Step-1 − One's compliment of 5 : 1010
Step-2 − Add 1 to above, giving 1011, which is -5

19. What is a static variable?
A static local variables retains its value between the function call and the default value is 0. The following function will print 1 2 3 if called thrice.

```
void f() {
staticint i;
  ++i;
printf("%d ",i);
}
```

20. When should we use the register storage specifier?
If a variable is used most frequently then it should be declared using register storage specifier, then possibly the compiler gives CPU register for its storage to speed up the look up of the variable.

21. What is a variable?

A variable is the name storage.

22. Which operator can be used to determine the size of a data type or variable?

Sizeof.

23.
What are valid operations on pointers?
The only two permitted operations on pointers are

Comparision ii) Addition/Substraction (excluding void pointers)

24. What are the basic Datatypes supported in C Programming Language?
Ans: The Datatypes in C Language are broadly classified into 4 categories. They are as follows:

1. Basic Datatypes
2. Derived Datatypes
3. Enumerated Datatypes
4. Void Datatypes

The Basic Datatypes supported in C Language are as follows:

Datatype Name	Datatype Size	Datatype Range
short	1 byte	-128 to 127
unsigned short	1 byte	0 to 255
char	1 byte	-128 to 127
unsigned char	1 byte	0 to 255
int	2 bytes	-32,768 to 32,767
unsignedint	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long	4 bytes	0 to 4,294,967,295
float	4 bytes	3.4E-38 to 3.4E+38
double	8 bytes	1.7E-308 to 1.7E+308
long double	10 bytes	3.4E-4932 to 1.1E+4932

25. What do you mean by Dangling Pointer Variable in C Programming?

Ans: A Pointer in C Programming is used to point the memory location of an existing variable. In case if that particular variable is deleted and the Pointer is still pointing to the same memory location, then that particular pointer variable is called as a Dangling Pointer Variable.

26. What do you mean by Dangling Pointer Variable in C Programming?

Ans: A Pointer in C Programming is used to point the memory location of an existing variable. In case if that particular variable is deleted and the Pointer is still pointing to the same memory location, then that particular pointer variable is called as a Dangling Pointer Variable.

27. Differentiate between calloc() and malloc()

Ans: calloc() and malloc() are memory dynamic memory allocating functions. The only difference between them is that calloc() will load all the assigned memory locations with value 0 but malloc() will not.

28.Can a C program be compiled or executed in the absence of a main()?

Ans: The program will be compiled but will not be executed. To execute any C program, main() is required.

29. What is a C Token?

Ans: Keywords, Constants, Special Symbols, Strings, Operators, Identifiers used in C program are referred to as C Tokens.

30. What is the difference between declaring a header file with <>and ” “?

Ans: If the Header File is declared using <> then the compiler searches for the header file within the Built-in Path. If the Header File is declared using ” ” then the compiler will search for the Header File in the current working directory and if not found then it searches for the file in other locations.