TCS Previous Papers

Technical and C Questions

TCS Previous Year Papers
and Study materials

Some Questions don't have options since we couldn't find the options while trying to get
questions from students who have already given TCS Test but anyways we have put the
questions so can at least give them a shot.

Q.A pointer variable can be
1. Changed within function.
2. Assigned an integer value.
3. None of these
4. Passed to a function as argument.
Correct Op: 4

Q. Which of the following uses structure?
1. Linked Lists
2. Array of structures
3. All of these
4. Binary Tree
Correct Op: 3

Q. Strings are character arrays. The last index of it contains the null-terminated character
1. \t
2. \1
3. \0
4. \n
Correct Op: 3

Q. Which of the following is a collection of different data types?
1. String
2. Structure
3. Array
4. Files
Correct Op: 2
Q. What function should be used to free the memory allocated by calloc() ?
1. free();
2. malloc(variable_name, 0)
3. dealloc();
4. memalloc(variable_name, 0)
Correct Op: 1

Q. In the standard library of C programming language, which of the following header file is designed for basic mathematical operations?

1. conio.h
2. stdio.h
3. math.h
4. Dos.h
Correct Op: 3

Q. int **ptr; is?
1. Pointer to integer
2. None of these
3. Pointer to pointer
4. Invalid declaration
Correct Op: 3

Q8. Which of the following special symbol allowed in a variable name?
1. (underscore)
2. - (hyphen)
3. | (pipeline)
4. * (asterisk)
Correct Op: 1

Q9. All keywords in C are in
1. Uppercase letters
2. None of these
3. Lowercase letters
4. Camel Case letters
Correct Op: 3

Q10. What should the program below print?

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void myfunc(char** param){
++param;
}
int main(){
char* string = (char*)malloc(64);
strcpy(string, "hello_World");
myfunc(&string);
myfunc(&string);
printf("%s\n", string);
// ignore memory leak for sake of quiz
return 0;
}
```

1. hello_World
2. ello_World
3. lo_World
4. llo_World
Correct Op: 1

Q: What is the output of this 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
c) 5 junk junk
d) Compile time error
Correct op: D

Explanations

It would have been 5 5 5 if it were **m and not **p.

Q. Which of the following statements about stdout and stderr are true?
a) They both are the same
b) Run time errors are automatically displayed in stderr
c) Both are connected to the screen by default.
d) stdout is line buffered but stderr is unbuffered.
Correct Op: D

Explanation -

a) **False. b) Not by default. c) Not by default. d) True.**

Q: Given the below statements about C programming language:
1) main() function should always be the first function present in a C program file
2) all the elements of an union share their memory location
3) A void pointer can hold address of any type and can be typcasted to any type
4) A static variable hold random junk value if it is not initialised
Which of the above are correct statements?
A) 2,3
B) 1,2
C) 1,2,3
D) 1,2,3,4
Correct Op – A

Explanations
In a file you can write a function before main() - False
all the elements of an union share their memory location - True.
A void pointer can hold address of any type and can be typcasted to any type - True
Static value - False as value is 0
In C, if an object that has static storage duration is not initialized explicitly, then:
— if it has pointer type, it is initialized to a NULL pointer;
— if it has arithmetic type, it is initialized to (positive or unsigned) zero;

— if it is an aggregate, every member is initialized (recursively) according to these rules;
— if it is a union, the first named member is initialized (recursively) according to these rules.

Q If a function is defined as static, it means
A) The value returned by the function does not change
B) all the variable declared inside the function automatically will be assigned initial value of zero
C) It should be called only within the same source code / program file.
D) None of the other choices as it is wrong to add static prefix to a function
Correct Op: C

Access to static functions is restricted to the file where they are declared. Therefore, when we want to restrict access to functions, we make them static.

Q: Comment on the below while statement=
while (0 == 0) { }
A) It has syntax error as there are no statements within braces {}
B) It will run forever
C) It compares 0 with 0 and since they are equal it will exit the loop immediately
D) It has syntax error as the same number is being compared with itself
Correct Op: B

while(0==0) {} is equivalent to while(1) {}

1. What will happen if in a C program you assign a value to an array element whose subscript exceeds the size of array?
A. The element will be set to 0.
B. The compiler would report an error.
C. The program may crash if some important data gets overwritten.
D. The array size would appropriately grow.
Answer: Option C
Explanation:
If the index of the array size is exceeded, the program will crash. Hence "option c" is the correct answer. But the modern compilers will take care of this kind of errors.

2. What does the following declaration mean?
int (*ptr)[10];
A. ptr is array of pointers to 10 integers
B. ptr is a pointer to an array of 10 integers
C. ptr is an array of 10 integers
D. ptr is an pointer to array
Answer: Option B

3. In C, if you 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
D.Address of the last element of array
Answer: Option C
Explanation:
The statement 'C' is correct. When we pass an array as a function argument, the base address of the array will be passed.

4. What will be the output of the program ?

```
#include<stdio.h>
int main()
{
int a[5] = {5, 1, 15, 20, 25};
int i, j, m;
i = ++a[1];
j = a[1]++;
m = a[i++];
printf("%d, %d, %d", i, j, m);
return 0;
}
```

A.2, 1, 15
B.1, 2, 5
C.3, 2, 15
D.2, 3, 20
Answer: Option C
Explanation:
Step 1: int a[5] = {5, 1, 15, 20, 25}; The variable arr is declared as an integer array with a size of 5 and it is initiapzed to
a[0] = 5, a[1] = 1, a[2] = 15, a[3] = 20, a[4] = 25 .
Step 2: int i, j, m; The variable i,j,m are declared as an integer type.
Step 3: i = ++a[1]; becomes i = ++1; Hence i = 2 and a[1] = 2
Step 4: j = a[1]++; becomes j = 2++; Hence j = 2 and a[1] = 3.
Step 5: m = a[i++]; becomes m = a[2]; Hence m = 15 and i is incremented by 1(i++ means 2++ so i=3)
Step 6: printf("%d, %d, %d", i, j, m); It prints the value of the variables i, j, m
Hence the output of the program is 3, 2, 15

5. Is there any difference int the following declarations?

```
int fun(int arr[]);
int fun(int arr[2]);
```

A.Yes
B.No
Answer: Option B
Explanation:
No, both the statements are same. It is the prototype for the function fun() that accepts one integer array as an parameter and returns an integer value.

6. Are the expressions arr and &arr same for an array of 10 integers?
A.Yes

B.No
Answer: Option B
Explanation:
Both mean two different things. arr gives the address of the first int, whereas the &arr gives the address of array of ints.

7. Which of the fplowing statements should be used to obtain a remainder after dividing 3.14 by 2.1?
A.rem = 3.14 % 2.1;
B.rem = modf(3.14, 2.1);
C.rem = fmod(3.14, 2.1);
D.Remainder cannot be obtain in floating point division.
Answer: Option C
Explanation:
fmod(x,y) - Calculates x modulo y, the remainder of x/y.
This function is the same as the modulus operator. But fmod() performs floating point divisions.

8. What are the types of pnkages?
A.Internal and External
B.External, Internal and None
C.External and None
D.Internal
Answer: Option B
Explanation:
External pnkage-> means global, non-static variables and functions.
Internal pnkage-> means static variables and functions with file scope.
None pnkage-> means Local variable

9. Which of the fplowing special symbp allowed in a variable name?
A.* (asterisk)
B.| (pipepne)
C.-(hyphen)
D._(underscore)
Answer: Option D
Explanation:
Variable names in C are made up of letters (upper and lower case) and digits. The underscore character ("_") is also permitted. Names must not begin with a digit.
Examples of vapd (but not very descriptive) C variable names:
=> foo
=> Bar
=> BAZ
=> foo_bar
=> _foo42
=> _
=> QuUx

10. Is there any difference between following declarations?
1 : extern int fun();

2 : int fun();
A. Both are identical
B. No difference, except extern int fun(); is probably in another file
C. int fun(); is overrided with extern int fun();
D. None of these
Answer: Option B
Explanation:
extern int fun(); declaration in C is to indicate the existence of a global function and it is defined externally to the current module or in another file.
int fun(); declaration in C is to indicate the existence of a function inside the current module or in the same file.

Ques. What could be the output for following?

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

(A) 2
(B) -2
(C) 1
(D) Error
--2 is incorrect, // Invalid because lvalue is required to increment

Ques. Predict the output of following code:

```
main()
{
int i=-1;
-i; //No change in value of i
printf("%d,%d",i,-i);
}
```

(A) -1, 1
(B) -1, -1
(C) 1, 1
(D) 0, 1

Ques. Predict the output of following code:

```
main()
{
int var=20; // scope of the local variable is within function or block
printf("%d,",var); //outer block
{
int var=30; //Inner block
printf("%d",var);
}
```

}
(A)Error
(B) 20,30
(C) 20,20
(D) Garbage value

Predict the output of following code:

```
main()
{
int var=20 ; // scope of the local variable is within function or
block
printf("%d,",var); //outer block
{
int var=30; //Inner block
printf("%d,",var);
}
printf("%d",var); //again in outer block
}
```

(A)Error (B)
20,30,20
(C) 20,20,20
(D) Garbage value

Which among the following operator
has the right to left associativity?
(A). Arithmetic
(B). logical
(C). Relational
(D). Increment/Decrement
Note: among logical operators logical NOT has right to left associativity , i.e . !
Operator

Predict the output of following code:

```
main()
{
int x,a=10;
x=9*5+ 7/3 -6+a; //45+2-6+10 = 51 // 7/3 =2 int division
printf("%d",x);
}
```

(A). 51
(B). 51.5
(C). 31
(D). None of these

Predict the output of following code:

```
main()
{
int a=10,x;
x= a-- + ++a;
```

```
printf("%d",x);
}
```
(A). 19
(B). 20
(C). 22
(D). 23

Note : For a-- value 10 is used and a is reduced to 9 post decrement
for ++a value 9 is incremented and new value 10 is used pre increment
therefore : x = 10 + 10 = 20

Predict the output of following code:

```
main()
{
int a=10,x=20;
a=a++ + 10; // a = 10+10 first a is increment to 11 but
overwritten by 20
x=x++ + ++a; // x = 20 + 21 a is incremented first from 20 to 21
printf("%d,%d",a,x);
}
```
(A). 22,43
(B). 12,21
(C). 10,20
(D). 42,42
No option has correct answer.
Correct answer is 21,41

Predict the output of following code:

```
main()
{
int i=10,j=2,k=0,m;
m=++i&&++j&&++k; // m = 11 && 3 && 1 = 1
printf("%d%d%d%d",i,j,k,m);
}
```
a. 11,3,1,1
b. 11,2,0,1
c. 11,3,1,0
d. 10,2,0,1

Predict the output of following code:

```
main()
{
int i=10;
printf("%d,%d",++i,++i);
}
```
a. 11,12
b. 12,11
c. 10,11
d. 11,10

Predict the output of following code:

```
main()
{
int a,x=(2,3,4,5); //valid statement x = last value in list
// x = 5 during declaration list
should be specified inside ( )
a=1,2,3,4,5; // valid; a = 1; first value of the list is
assigned to variable
printf("%d%d",x,a);
}
```

a. Error b. 5,1
c. 2,1 d. 5,5

Predict the output of following code:

```
main()
{
int a,x=2,3,4,5; // Error because list is not within ( )
a=1,2,3,4,5;
printf("%d%d",x,a);
}
```

a. Error
b. 5,1
c. 2,1
d. 5,5

Predict the output of following code:

```
main()
{
int x=10,y=-10;
printf("%x \t",x<<2); // %X hexadeciamal value displayed
printf("%x\t",y>>2);
}
```

a. 28 fffd
b. 40 -3
c. Error
d. 0,1

Note : here 16 bit computer is considered.
In bitwise operations if number is +ve then simply specied number of bits shifted in specified direction L or R.
If Number is –VE then 2's complement is used
Therefore -10 = 1111 0110 = F5 (2's complement)
after 2positions shift → 1111 1101 → FD
in decimal → 1111 1101→10000 0011 → -03

Predict the output of following code:

```
main()
{
unsigned int a= -1;
signed int b=10;
```

```
if(a<b) // if( 10 < 255)
printf("a is the smallest number");
else
printf("b is the smallest number");
}
```

a. a is the smallest number
b. b is the smallest number
c. Error
d. Both statements

Note: //for unsigned variable negative valued is assigned computer takes 2'complement number to assign.
a = 1111 1111= FF (255)

Predict the output of following code:

```
main()
{
if(1) // True always
printf("hai");
else
printf("hello");
}
```

a. Error
b. hai
c. hello
d. No output

Predict the output of following code:

```
main()
{
if(5,4,3,2,1, 0 ) // Last value of list is considered
printf("True");
else
printf("False");
}
```

a. True
b. False
c. True False
d. Error

Predict the output of following code:

```
main()
{
if(5,4,3,2,1,0,8, 9 ) // Last value of list is considered
printf("True");
else
printf("False");
}
```

a. True
b. False

c. True False
d. Error

Predict the output of following code:

```
main()
{
if( printf(“Hai”)) // prints content of printf statement and
//takes length of the string for if case //execution in this case
its 4, So true
printf(“Face”);
else
printf(“Focus”);
}
```

a. Error
b. HaiFace
c. HaiFocus
d. Face

Predict the output of following code:

```
main()
{
if( printf(“”)) // prints content of printf statement and
//takes length of the string for if case //execution in this
case its 1, So true
printf(“Face”);
else
printf(“Focus”);
}
```

a. Error
b. Face
c. “”Focus
d. Face

Predict the output of following code:

```
main()
{
if(printf(“O”, printf(“Two”))) / prints content of printf statement and takes
length of //the string for if case execution in this case its
4, So true
printf(“Face”);
else
printf(“Focus”);
}
```

a. Error
b. OTwoFace
c. HaiFocus
d. Face

Predict the output of following code:

```
main()
{
if(-1) //2's complement its value FF
printf("True");
else
printf("False");
}
```

a. True
b. False
c. True False
d. Error

Predict the output of following code:

```
main()
{
int a=100,b=300;
if(a>50) // No braces, so only one immediate statement is part of if
a=200 ;
b=400;
printf("%d",b);
}
```

a. 400
b. 300
c. 100
d. Error

Find the error, if any, in the while loop

```
main()
{
int i = 1;
while( i < = 5)
{
printf("%d ", i);
if ( i < 2)
goto here ;
}
}
fun()
{
here:
printf("\n I am here");
}
```

Error: _Label → here used in main function but not defined

Predict the output of following code:

```
main()
{
int a=2;
if(a-- , --a, a) // if(2, 0, 0) last value 0 →False
```

```
printf(“Tom”);
else
printf(“Jerry”);
}
```

a. Tom
b. Jerry
c. Tom Jerry
d. Error

Predict the output of following code:

```
main()
{
int a=2;
switch(a)
{
case 1: printf(“one”);
case 2: printf(“Two”); // Executable code ; No break statement
case 3: printf(“Three”);
default:printf(“Invalid option”);
}
}
```

a. onetwothree
b. Invalid option
c. one two
d. None of these

Guess the output:

```
main()
{
printf(“%d”, sizeof('a')); //same as → sizeof(97)
}
```

a. 2 or 4
b. 1 or 3
c. Garbage value
d. ASCII value of a

NOTE:
// sizeof takes ascii value of character and determines number of bytes required by it. Ascii is number, Number is of type int. so integer requires either 2 in 16 or 4 in 32 bit machine

Predict the output of following code:

```
main()
{
int a=b=c=d=10; // error: ‘b’ , ‘c’, ‘d’ undeclared
printf(“%d,%d,%d,%d”,a,b,c,d);
}
```

a. Error
b. 10,10,10,10
c. GV,GV,GV,10

d. GV,GV,GV,GV
NOTE: GV-Garbage Value

Predict the output of following code:

```
main()
{
int b,c,d;
int  a=b=c=d=10;
printf("%d,%d,%d,%d",a,b,c,d);
}
```

a. Error
b. 10,10,10,10
c. GV,GV,GV,10
d. GV,GV,GV,GV
NOTE: GV-Garbage Value

Predict the output of following code:

```
main()
{
int  sum;
char ch1='a';
char ch2='b';
sum=ch1+ch2; // ascii sum; sum = 97+98 = 195
printf("%d",sum);
}
```

a. Error
b. 195
c. 201
d. "ab"

Predict the output of following code:

```
main()
{
float a=1.1;
double b=1.1;
if(a==b) // datatype is different cant be compared; hence result will be 0
printf("equal");
else
printf("not equal");
}
```

a. equal
b. not equal
c. Error
d. equal not equal

What is the output for following?

```
main()
{
printf("%%%%");
```

}
a. %%%%
b. %%
c. Error
d. Garbage Value
Note:
A `%' is written.
No argument is converted.
The complete conversion specification is`%%'.
so, “%%%%” → prints → %%

```
main()
{
printf(“%d”);
}
```

a.0
b. 1
c. Garbage value
d. Error

Guess the output:

```
main()
{
printf(“\n ks”);
printf(“\b mi \a”);
printf(“\r ha \n”);
}
```

a. ksmiha
b. mis
c. hai
d. hamiks
Note:
after 1 st statement execution:
ks
After 2 nd : k mi
After 3 rd : ha i

Predict the output of following code:

```
main()
{
100; // valid but no effect
printf(“%d”,100);
}
```

a. Error
b. 100
c. Garbage value
d. 100100

Predict the output of following code:

```
main()
```

```
{
printf(“ %d ”,printf(“ FACE ”)); //valid
}
```

a. FACE4
b. Error
c. Garbage value
d. FACE

Note:
First prints content of printf statement
Then prints its length in outer printf statement

Predict the output of following code:

```
main()
{
printf(“%d”,printf(“FACE”)); //valid
printf("",printf("Third "),printf("Second "),printf("First "));
printf("%f%f%f",printf("Third "),printf("Second "),printf("First "));
printf("%d%d%d",printf("Third "),printf("Second "),printf("First "));
}
```

Note: Output
FACE4
First Second Third
First Second Third 0.000000 0.000000 0.000000
First Second Third 676

Predict the output of following code:

```
main()
{
printf(“FACE”+2); // valid skip specified number of
//characters from start
}
```

a.FACE
b. FA
c. CE
d. Garbage value

Predict the output of following code:

```
main()
{
int a=2000;
printf(“%2d”,a); //format specification
}
```

a. 2000
b. 20
c. 4000
d. Garbage value

Predict the output of following code:

```
main()
```

```
{
int x,a=10;
x=a==10?printf("hai\t"):printf("hello\n");
printf("%d",x);
}
```

a. hai 4
b. Error
c. hello 3
d. hai hello

Note:
First prints content of printf for the case true
Then printf its length stored in x

Predict the output of following code:

```
main()
{
int a=10,b=20,c=5,d;
d=a<b<c; // d = (10 < 20) < 5 = 1 < 5 = 1
printf(“%d”,d);
}
```

a.0
b. 1
c. 5
d. Error

How many of the following are invalid variable name?
NUMBER _num 93num num93
first.name last name nUMBER
midname. 4321
a.5
b. 3
c. 2
d. More than 5

Predict the output

```
int main()
{
float f=5,g=10;
enum{i=10,j=20,k=50};
printf("%d\n",++k); //k is enumerated constant
printf("%f\n",f<<2); // f is float
printf("%lf\n",f%g); //invalid operands floats
printf("%lf\n",fmod(f,g)); // function is in math.h
return 0;
}
```

Output: Errors

What would the output of this program be?

Will there be any error?

```
#define a 10
int main()
{
#define a 50 //Warning is raised for redefining a
printf("%d",a); //Prints new value of a i.e. 50
getchar(); //waits for character input
return 0;
}
```

Output: 50

Predict the output of below program.

```
int main()
{
char arr[] = "PrepInsta";
printf("%d", sizeof(arr));
getchar();
return 0;
}
```

Output: 9

Predict the output

```
#include <stdio.h>
int main(void)
{
http://prepinsta.com/ //valid specification http://
printf("Hello, World !!!\n");
return 0;
}
```

Output: Hello, World !!!

Predict the output

```
#include <stdio.h>
int main(void)
{
int x = printf("PrepInsta"); //first prints message then assigns length its to x
printf("%d", x);
return 0;
}
```

(A) PrepInsta9
(B) PrepInsta10
(C)PrepInstaPrep
(D) PrepInsta1

Output of following program?

```
#include<stdio.h>
int main()
{
printf("%d", printf("%d", 1234));
```

```
return 0;
}
```
(A) 12344
(B) 12341
(C) 11234
(D) 41234
Note: First prints inner message 1234
then its length in outer printf 4
Hence output : 12344

Output of following program?
```
#include<stdio.h>
int main()
{
printf("%d", printf("%d", printf(“%d”,543210)));
return 0;
}
```
(A) 54321061
(B) 543210
(C) 5432101
(D) 5432106
Note: First prints inner message 543210
then its length in outer printf 6
then its length in outer printf 1
Hence output : 54321061

Predict the output
```
#include <stdio.h>
int main()
{
float c = 5.0;
printf ("Temperature in Fahrenheit is %.2f", (9/5)*c + 32);
return 0;
}
```
(A) Temperature in Fahrenheit is 41.00
(B) Temperature in Fahrenheit is 37.00
(C) Temperature in Fahrenheit is 0.00
(D) Compiler Erro r
Note: 9/5 int value → 1*5+32 =37.00 (float)

What will be output of the following program?
```
#include<stdio.h>
int main(){
int a=2,b=7,c=10;
c=a==b; //assign 0 to c
printf("%d",c);
return 0;
}
```
Output: 0

What will be output of the following program?

```
#include<stdio.h>
void main(){
int x;
x=10,20,30; // Assigns first value to z i.e. 10
printf("%d",x);
return 0;
}
```

Output: 10

What will be output of the following program?

```
#include<stdio.h>
int main(){
int a=0,b=10;
if(a=0){ //value a is 0 hence else case is executed
printf("true");
}
else{
printf("false");
}
return 0;
}
```

Output: false

```
int main()
{
signed char i=0;
for(; i >= 0; i++)
printf("%d\n", i);
getchar();
return 0;
}
```

Output: prints ascii numbers from 0 to 127

What is the output of this C code?

```
#include <stdio.h>
void main()
{
double k = 0;
for (k = 0.0; k < 3.0; k++)
printf("Hello");
}
```

a) Run time error
b) Hello is printed thrice
c) Hello is printed twice
d) Hello is printed infinitely

Find the output

```
# include <stdio.h>
int main()
{
int i=0;
for(i=0; i<20; i++)
{
switch(i)
{
case 0: i+=5;
case 1: i+=2;
case 5: i+=5;
default: i+=4;
break;
}
printf("%d ", i);
}
getchar();
return 0;
}
```

Output : 16 21 //start 0+5+2+5+4 = 16 for first iteration
// i++ in for loop changes i=17 then default case 17+4=21 ends loop

Predict the output of following code:

```
main()
{
int a[20]={1,2,3};
printf(“%d”,sizeof(a));
}
```

a. 20
b. 6
c. Error
d. 40 or 80

Output: a has 20 memory location of each 2 bytes in case 16 bit m/c
20*2 = 40 bytes or 20 * 4 = 80 in case 32 bit m/c

Predict the output of following code:

```
main()
{
int a[]={1,5};
printf("%d,",*a); //a is base address i.e. a[0] *a its content
(*a)++; //content of a[0] is incremented by 1
printf("%d",*a); //new value will be 2
}
```

a.1, 5
b. 1, 2
c. Error
d. 1, 6

Predict the output of following code:

```
main()
{
int a[3]={1,2,3};
printf("%d", 2[a]); // 2[a] → a[2] → *(a+2) all are same
}
```

a.3
b. 2
c. Error
d. 6

Predict the output of following code:

```
main()
{
printf("%c",2["hai"]); // 2 nd character in the string i.e. i
}
```

a. 2
b. h
c. Error
d. I

h→0
a→1
i→2

Predict the output of following code:

```
main()
{
char s[]= ""; //null character
printf("%d",sizeof(s));
}
```

a.1
b. Garbage
c. Error
d. 2

Predict the output of following code:

```
main()
{
int i=0,n;
int a[10]={1,2,3,4,5,6,7,8};
n=a[++i]+ i++ + a[i++] + a[i];
printf("%d",n);
}
```

a. 7
b. 10
c. 9
d. 8

Output: n = a[1] + 1 + a[2] + a[3] = 2 + 1 + 3 + 4 = 10

Point out the Error :

```
main()
{
int a[][]={{1,2},{3, 4 },{5,6}};
printf("%d",a[1][1]);
}
```
Output: 4

Predict the output of following code:
```
main()
{
int a[3][2]={{1,2},{3,4},{5,6}};
printf("%d,%d,%d\n",a[2][1],*(a[2]+1),*(*(a+2)+1));
}
```
a. Error
b. Garbage value
c. 2, 3, 4
d. 6, 6, 6
Output: a[2][1] = 6 *(a[2]+1)= a[2][1] =6
((a+2)+1) = *(address of row + 1) = content of row 2 col 1 = 6

Predict the output of following code:
```
main()
{
int arr2D[3][3]={1,2,3,4,5,6};
printf("%d\n", ((arr2D==* arr2D)&&(* arr2D == arr2D[0])) );
}
```
a. Error
b. 1
c. 0
d. 6
Note: base address of the array is compared

Predict the output :
```
main()
{
int a=10,*p;
int *vp;
p=&a;
vp=p;
printf("%d",*p); // p→ a both p & vp points to a
printf("%d",*vp); // vp → a
}
```
a . 1010
b. Error type casting required
c. 10 Garbage values
d. Both are garbage value

Predict the output of following code:
```
main()
```

```
{
char a= ‘a’,*p;
p=&a;
printf(“%d,%d”,sizeof(*p) , sizeof(p));
}
```

a. 1, 2
b. 2, 2
c. 1, 1
d. Error
Note: sizeof(p) is 2 for 16 bit m/c and 4 for 32 bit m/c

Predict the output of following code:

```
main()
{
int a=10,*p;
p=&a;
printf("%d",*&*p);
}
```

a.10
b. address of a
c. Error
d. Address of p

Predict the output of following code:

```
main()
{
char *str= “face”,b;
printf(“%d”,-2[str]); // printf ascii value of c with -
}
```

a.-99
b. c
c. Error
d. b

Predict the output of following code:

```
main()
{
int i=10,j=20;
int *p,*q;
*p=i; // only value is assigned to *p
q=&j; //pointer initialization
printf(“%d,%d”,*p,*q);
}
```

a.10, 10
b. 10, 20
c. Error
d. Garbage value, 20

```
char ** array [12][12][12];
```

Consider array, defined above. Which one of the following definitions and initializations of p is valid?

a. char ** (* p) [12][12] = array;

b. char ***** p = array;

c. char * (* p) [12][12][12] = array;

d. const char ** p [12][12][12] = array;

Note : array is pointer to pointer array

( ) has higher priority so *p is pointer to pointer to pointer array

What is the difference between int *arr[10]; and int (*arr)[10];

int *arr[10]; // declares array of 10 pointers

int (*arr)[10]; // pointer to array 10 intergers ;( ) higher priority

Find the output:

```
void main()
{
int i=7;
printf("%d",i++*i++); // 7*8
}
```

a. 49 b. 56 c. 64 d. Garbage value

Find the error/output.

```
#include<stdio.h>
int i; // i =0; initialized by default value
int kunfu();
int main()
{
while(i)
{
kunfu();
main();
}
printf("Lovely\n"); //only executes
return 0;
}
int kunfu()
{
printf("Pretty");
}
```

a.Lovely

b. Pretty

c. Lovely Pretty

d. Infinite loop

Find the error/output:

```
#include<stdio.h>
int calc(int);
int main()
{
int a, b;
```

```
a = calc(123);
b = calc(123);
printf("%d, %d\n", a, b);
return 0;
}
int calc(int n)
{
int s, d;
if(n!=0)
{
d = n%10;
n = n/10;
s = d+calc(n);
}
else
return 0;
return s;
}
```

a.6 6

b. 4 4

c. 2 8

d. Compilation error

Find out error/output:

```
f(int p , int q)
{
int p;
p = 5;
return p;
}
```

a. 5

b. 0

c. Missing parenthesis in return statement

d. Error: re-declaration of p

Find the error/output:

```
#include <stdio.h>
float sub(float, float);
int main()
{
float a = 4.5, b = 3.2, c;
c = sub(a, b);
printf("c = %f\n", c);
return 0;
}
float sub(float a, float b) //function not declared
{
return (a - b);
}
```

a. 2
b. Compilation error
c. 1.300000
d. Garbage value

Predict the output of following code:

```
struct student
{
int stuid=1234;
char stuname[5]= “abcde”; //invalid can’t initialize members
}s1;
main()
{
struct student s1;
printf( “%d,%s”,s1.stuid,s1.stuname);
}
```

a. 1234,abcde
b. 12341234
c. Error
d. abcdeabcde

Predict the output of following code:

```
struct employee
{
int  empid;
float empbasic;
}emp1={13}; //valid initialization first value 13 second value set 0 by default
main()
{
struct employee ;
printf(  “%d,%f”,emp1.empid,emp1.empbasic);
}
```

a. 13, 13
b. 13,0.000000
c. Error
d. 13, garbage value

Point out the error in the following code :

```
main()
{
struct mystruct
{
int a;
mystruct b; // structure must define before use
mystruct *p;
};
}
```

a. Error in pointer declaration
b. Error in variable ‘b’ declaration

c. No Error
d. both a and b

Predict the output of following code:

```
struct student
{
int stuid;
char stuname[5];
}s1={1234,“abcde”};
main()
{
struct student s2={5678, “abcde”};
if(s1==s2) // invalid comparision struct instances
printf(“true”);
else
printf(“false”);
}
```

a.true
b. false
c. Error
d. truefalse

Predict the output of following code:

```
struct birthdate
{
int date;
int month;
int year;
};
main()
{
struct student
{
int stuid;
char stuname[20];
struct birthdate dob;
} s1={1234, “abcde”};
printf(“%d”,sizeof(s1)); / / sizeof(int)+sizeof(stuname)+sizeof(birthdate)
}
```

a.13
b. 28
c. Error
d. 7

Predict the output of following code:

```
union temp
{
int m1;
```

```
char ch;
};
main()
{
union temp u1;
u1.m1=10;
u1.ch=20 ; // union uses common memory so recent value is present
printf(“%d%d”,u1.m1,u1.ch);
}
```

a.10, 10
b. 2020
c. Error
d. 1020

Predict the output of following code:

```
struct birthdate
{
int date;
int month;
int year;
};
main()
{
union student
{
int stuid;
char stuname[2];
struct birthdate dob; // dob require larger memory hence max size of ul is 6
} u1={1234, “ab”};
printf(“%d”,sizeof(u1));
}
```

a.10
b. 28
c. Error
d. 6

Predict the output of following code:

```
struct mystruct
{
int x; // default value of members of structure is 0
int y;
};
struct mystruct s1,*pp;
main()
{
pp=&s1;
printf("%d%d\n",(*pp).x,(*pp).y);
printf("%d%d\n",pp->x,pp->y);
}
```

a. 00
b. Garbage values
c. Error
d. 11

Choose the correct option to print out a & b:

```
#include<stdio.h>
float a;
double b;
```

a. printf("%f %lf", a, b)
b. printf("%f %Lf", a, b);
c. printf("%Lf %f", a, b);
d. None

Find the output:

```
#include<stdio.h>
int main()
{
float a=2.15529;
printf("%2.1f\n", a); //rounds decimal value to next higher digit
return 0;
}
```

a.2.15
b. 2.1
c. 2.2
d. Error

2) Where the local variables are stored?
A. Disk
B. Stack
C. Heap
D. 13

Select the missing statement?

```
#include<stdio.h> long int fact(int n); int main()
{
\\missing statement }
long int fact(int n) {
if(n>=1)
return n*fact(n-1);
else
return 1;
}
```

Options
A. printf(“%ll\n",fact(5));
B. printf("%u\n",fact(5));
C. printf("%d\n",fact(5));
D. printf("%ld\n",fact(5));

Which of the following indicate the end of the file?
A. Feof()
B. EOF
C. Both feof() and EOF
D. None of the mentioned

If a function's return type is not explicitly defined then it's default to_____(In C).
A. int
B. float
C. void
D. Error

For passing command line argument the main function should be like _____
A. int main(char *argv[], int argc)
B. int main(int argc)
C. int main(char *argv[])
D. int main(int argc, char *argv[])

How many times the below loop will be executed?
#include<stdio.h> int main()
{
int i; for(i=0;i<5;i++) {
printf("Hello\n"); }
}
Options
A. 5
B. 1
C. 0
D. 3

Which of the following is a User-defined data type?
A. long int
B. double
C. unsigned long int
D. enum

Which has the highest precision?
A. float
B. double
C. unsigned long int
D. Long int

The following table provide the details of standard floating-point types with storage sizes
and value ranges and their precision –
Type Storage size Value range Precision
float
4 byte

1.2E-38 to 3.4E+38 6 decimal places
double
8 byte
2.3E-308 to 1.7E+308
15 decimal places
long double
10 byte
3.4E-4932 to 1.1E+4932
19 decimal places

What will be the output/error?(for input: 6, 9

```
#include<stdio.h> int fg(int,int);
int main()
{
int n1,n2,g;
scanf("%d%d", &n1,&n2);
g=fg(n1,n2);
printf("%d",g); }
int fg(int x,int y)
{
while(x!=y) {
if(x>y)
return fg(x-y,y);
else
return fg(x,y-x);
}
return x; }
```

Options
A. 3
B. 6
C. 9
D. Error

How to dynamically release memory?
AnsA. With Free Statement
Free()
Truncate()
delete()
release()

Ques. What is the function of ftell?
Ans. To get the current file Position
To get the current file name
To get the current file Position
To get the current file attributes
To get the current file status
Till Page 46 F

Question 1: Use of an increment statement or decrement statement in C?

Answer:
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.
Two types of increments are:
1. pre increment: (increment by 1 then print) and
2. post increment: (print then incremented value will be in buffer). Same thing will be with decrement.

Question 2: In programs we place comment symbols on some codes instead of deleting it. How does this aid in debugging?

Answer:
Placing comment symbols /* */ around a code, also referred to as commenting out , is a way of isolating some codes that you think maybe causing errors in the program, without deleting the code.

Question 3: What is the use of a '\0' character?

Answer:
This character is used primarily to show the end of a string value.

Question 4: What is the difference between the = symbol and == symbol?

Answer:
The = symbol is often used in mathematical operations. It is used to assign a value to a given variable. On the other hand, the == symbol, also known as equal to or equivalent to, is a relational operator that is used to compare two values.

Question 5: In C Programming, which of the following operators is incorrect and why? (>=, <=, <>, ==)

Answer:
<> is incorrect, all other operators are relational operators. While this operator is correctly interpreted as not equal to in writing conditional statements, it is not the proper operator to be used in C programming. Instead, the operator != must be used to indicate not equal to condition.

Question 6: Can the curly brackets { } be used to enclose a single line of code?

Answer:
While curly brackets are mainly used to group several lines of codes, it will still work without error if you used it for a single line. Some programmers prefer this method as a way of organizing codes to make it look clearer, especially in conditional statements.

Question 7: Can I use int data type to store the value 32768? Why/why not?

Answer:
No. int data type is capable of storing values from -32768 to 32767. To store 32768, you can use long int instead. You can also use 'unsigned int, assuming you don't intend to store negative values.

Question 8: Can two or more operators such as \n and \t be combined in a single

line of program code?

Answer: Yes, it's perfectly valid to combine operators, especially if the need arises.

For example: you can have a code like 'printf ('Hello\n\n\'World\')' to output the text 'Hello' on the first line and 'World' enclosed in single quotes to appear on the next two lines.

Question 9: When is the 'void' keyword used in a function?

Answer:

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.

Question 10: Write a loop statement that will show the following output:

1
12
123
1234
12345

Answer:

```
for (a=1; a<=5; i++) {
for (b=1; b<=a; b++)
printf("%d",b);
printf("\n");
}
```

Question 1: How would you round off a value from 1.66 to 2.0?

A. ceil (1.66)
B. floor (1.66)
C. roundup (1.66)
D. Round to (1.66)

Answer: A

```
/* Example for ceil() and floor() functions: */
#include<stdio.h>
#include<math.h>
int main()
{
printf("\n Result : %f" , ceil(1.44) );
printf("\n Result : %f" , ceil(1.66) );
printf("\n Result : %f" , floor(1.44) );
printf("\n Result : %f" , floor(1.66) );
return 0;
}
// Output:
// Result : 2.000000
// Result : 2.000000
```

// Result : 1.000000
// Result : 1.000000

Question 2: What will be the output of the program?

```
#include<stdio.h>
int X=40;
int main()
{
int X=20;
printf("%d\n", X);
return 0;
}
```

A.20
B.40
C.Error D.No
Output
Answer: A
Whenever there is conflict between a local variable and global variable, the local variable gets priority.

Question 3: A long double can be used if range of a double is not enough to accommodate a real number.
A. True
B. False
Answer: A
True, we can use long double; if double range is not enough.
Double = 8 bytes.
Long double = 10 bytes.

Question 4: A float is 4 bytes wide, whereas a double is 8 bytes wide.
A.True
B. False
Answer: A
True,
float = 4 bytes.
Double = 8 bytes.

Question 5: If the definition of the external variable occurs in the source file before its use in a particular function, then there is no need for an extern declaration in the function.
A. True
B. False
Answer: A
True, when a function is declared inside the source file, that function (local function) get a priority than the extern function. So there is no need to declare a function as extern inside the same source file

Question 6: If the definition of the external variable occurs in the source file before its use in a particular function, then there is no need for an extern

declaration in the function.

A. True
B. False
Answer: A
True, When a function is declared inside the source file, that function(local function) get a priority than the extern function. So there is no need to declare a function as extern inside the same source file

Question 7: Size of short integer and long integer can be verified using the size of() operator.

A. True
B. False
Answer: A
True, we can find the size of short integer and long integer using the sizeof() operator.

Question 8: Range of double is -1.7e-38 to 1.7e+38 (in 16 bit platform - Turbo C under DOS)

A. True
B. False
Answer: B
False, the range of double is -1.7e-308 to 1.7e+308.

Question 9: Size of short integer and long integer would vary from one platform to another.

A. True
B. False
Answer: A
True, Depending on the operating system/compiler/system architecture you are working on, the range of data types can vary.

Question 10: Range of float id -2.25e-308 to 2.25e+308

A. True
B. False
Answer: Option B
False, the range of float is -3.4e-38 to 3.4e+38.

Question 1: What is wrong in this statement?
scanf(%d,whatnumber);

Answer:
An ampersand '&' symbol must be placed before the variable name whatnumber. Placing & means whatever integer value is entered by the user is stored at the address of the variable name. This is a common mistake for programmers, often leading to logical errors.

Question 2: What does the format %10.2 mean when included in a printf statement?

Answer:
This format is used for two things: to set the number of spaces allotted for the

output number and to set the number of decimal places. The number before the decimal point is for the allotted space, in this case it would allot 10 spaces for the output number. If the number of space occupied by the output number is less than 10, addition space characters will be inserted before the actual output number. The number after the decimal point sets the number of decimal places, in this case, it's 2 decimal spaces.

Question 3: What are linked list?

Answer:
A linked list is composed of nodes that are connected with another. In C programming, linked lists are created using pointers. Using linked lists is one efficient way of utilizing memory for storage.

Question 4: What are binary trees?

Answer:
Binary trees are actually an extension of the concept of linked lists. A binary tree has two pointers, a left one and a right one. Each side can further branch to form additional nodes, which each node having two pointers as well.

Question 5: Differences between C and Java?

Answer:
JAVA is Object-Oriented while C is procedural.
2. Java is an Interpreted language while C is a compiled language.
3. C is a low-level language while JAVA is a high-level language.
4. C uses the top-down approach while JAVA uses the bottom-up approach.
5. Pointer goes backstage in JAVA while C requires explicit handling of pointers.

Question 6: In header files whether functions are declared or defined?

Answer: Functions are declared within header file. That is function prototypes exist in a header file, not function bodies. They are defined in library (lib).

Question 7: What are the different storage classes in C?

Answer:
There are four types of storage classes in C. They are extern, register, auto and static.

Question 8: What does static variable mean?

Answer:
Static is an access qualifier. If a variable is declared as static inside a function, the scope is limited to the function, but it will exists for the life time of the program. Values will be persisted between successive calls to a function.

Question 9: How do you print an address?

Answer:
Use %p in printf to print the address.

Question 10: What are macros? What are its advantages and disadvantages?

Answer:
Macros are processor directive which will be replaced at compile time.

The disadvantage with macros is that they just replace the code they are not function calls. Similarly the advantage is they can reduce time for replacing the same values.

Question 1: Difference between pass by reference and pass by value?

Answer:
Pass by value just passes the value from caller to calling function so the called function cannot modify the values in caller function. But Pass by reference will pass the address to the caller function instead of value if called function requires to modify any value it can directly modify.

Question 2: What is an object?

Answer:
Object is a software bundle of variables and related methods. Objects have state and behaviour.

Question 3: What is a class?

Answer:
Class is a user-defined data type in C++. It can be created to solve a particular kind of problem. After creation the user need not know the specifics of the working of a class.

Question 4: What is the difference between class and structure?

Answer:
Structure: Initially (in C) a structure was used to bundle different type of data types together to perform a particular functionality. But C++ extended the structure to contain functions also.
The major difference is that all declarations inside a structure are by default public.
Class: Class is a successor of Structure. By default all the members inside the class are private.

Question 5: What is pointer?

Answer:
Pointer is a variable in a program is something with a name, the value of which can vary. The way the compiler and linker handles this is that it assigns a specific block of memory within the computer to hold the value of that variable.

Question 6: What is the difference between null and void pointer?

Answer:
A Null pointer has the value 0. Void pointer is a generic pointer introduced by ANSI. Generic pointer can hold the address of any data type.

Question 7: what is function overloading?

Answer:
Function overloading is a feature of C++ that allows us to create multiple functions with the same name, so long as they have different parameters.Consider the following function:
int Add(int nX, int nY)

```
{
return nX + nY;
}
```

Question 8: what is friend function?

Answer:

A friend function for a class is used in object-oriented programming to allow access to public, private, or protected data in the class from the outside. Normally, a function that is not a member of a class cannot access such information; neither can an external class. Occasionally, such access will be advantageous for the programmer. Under these circumstances, the function or external class can be declared as a friend of the class using the friend keyword

Question 9: What do you mean by inline function?

Answer: The idea behind inline functions is to insert the code of a called function at the point where the function is called. If done carefully, this can improve the application's performance in exchange for increased compile time and possibly (but not always) an increase in the size of the generated binary executables.

Question 10: Tell me something about abstract classes?

Answer:

An abstract class is a class which does not fully represent an object. Instead, it represents a broad range of different classes of objects. However, this representation extends only to the features that those classes of objects have in common. Thus, an abstract class provides only a partial description of its objects.

Question 1: What is the difference between an array and a list?

Answer:

Array is collection of homogeneous elements. List is collection of heterogeneous elements.
For Array memory allocated is static and continuous. For List memory allocated is dynamic and random.
Array: User need not have to keep in track of next memory allocation.
List: User has to keep in Track of next location where memory is allocated.
Array uses direct access of stored members; list uses sequential access for members.

Question 2: What are the differences between structures and arrays?

Answer:

Arrays are a group of similar data types but Structures can be group of different data types.

Question 3: What is data structure?

Answer:

A data structure is a way of organizing data that considers not only the items stored, but also their relationship to each other. Advance knowledge about the relationship between data items allows designing of efficient algorithms for the manipulation of data.

Question 4: Can you list out the areas in which data structures are applied extensively?
Answer:
Compiler Design,
Operating System,
Database Management System,
Statistical analysis package,
Numerical Analysis,
Graphics,

Question 5: What are the advantages of inheritance?
Answer:
It permits code reusability. Reusability saves time in program development. It encourages the reuse of proven and debugged high-quality software, thus reducing problem after a system becomes functional.

Question 6: Advantages of a macro over a function?
Answer:
Macro gets to see the Compilation environment, so it can expand #defines. It is expanded by the pre-processor.

Question 7: What is command line argument?
Answer:
Getting the arguments from command prompt in c is known as command line arguments. In c main function has three arguments. They are:
Argument counter
Argument vector
Environment vector

Question 8: What are the 4 basics of OOP?
Answer:
Abstraction, Inheritance, Encapsulation, and Polymorphism.

Question 9: Tell how to check whether a linked list is circular.
Answer:
Create two pointers, each set to the start of the list. Update each as follows:
```
while (pointer1) {
pointer1 = pointer1->next;
pointer2 = pointer2->next; if (pointer2) pointer2=pointer2->next;
if (pointer1 == pointer2) {
print ("circular\n");
}
}
```

Question 10: Write a program to swap two numbers without using a temporary variable.
Answer:
```
void swap(int &i, int &j)
{
```

```
i=i+j;
j=i-j;
i=i-j;
}
```

Ques. 1 Which is the character array used to accept command line arguments?
A) char argv
B) char* argv[]
C) char argv[]
D) char* argv

Ques. 2 What is a dangling pointer?
Points to garbage value
Points to a function
Both a and b
None

Ques. 3 Which is not a string function?
A) strstr
B)strcmp
C) strupr
D) strchr

Ques. 4 Which of the following does not require to include math.h header file?
A) pow()
B) rand()
C)sqrt()
D) sinh()

Ques. 5 What is the task of pre-processor?
A) Expanding
B) Compiling
C) Linking
D) All of the above

Ques. 6 Which of the following is true?
A) realloc() can change the memory size of arrays
B) Unary operator works on only one operand
C) Struct and Union works in same way.
D) None of the above

Ques. 7 Which of this is used to skip one iteration:
A) break
B) continue
C) goto
D) return

Ques. 8 Which address does a pointer to an array store:
A) Memory address of the first element of the array Don't remember the other

options.

Ques. 9 Predict the output:

```
float a = 0.1;
if(a==0.1)
printf("Yes");
else
printf("No");
```

Answer would be No.

Ques. 10 Another output based question which basically displayed the input string in reverse pattern.

For example, ABRACADABRA was displayed as ARBADACARBA.