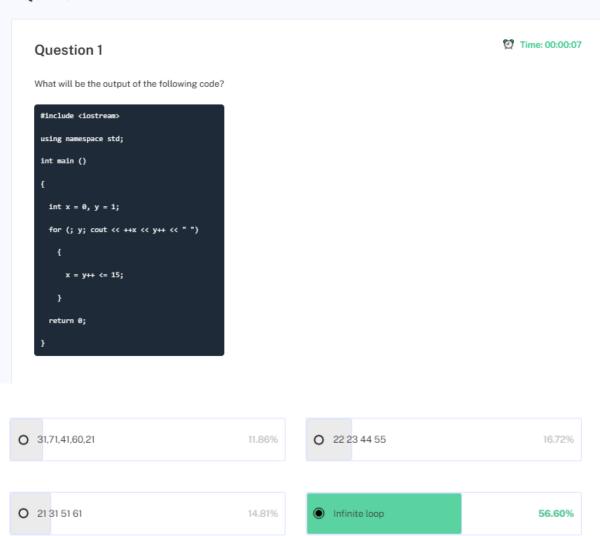
Quiz-6



The code will get into an infinite loop after entering in the for loop, and it will keep printing the value of y

Question 2 Time: 00:00:03

What will be the output of the following code?

```
#includecstdio.h>
int main ()
{
    int i = 1, j;
    for (;;)
    {
        i++;
        if (i)
            j = --i;
        if (i < 5)
            printf ("Royal Pass ", i++);
        else
            break;
    }
    return 0;
}</pre>
```

O will print Royal Pass 4 times

- O No, compile error but it will run into an infinite loop printing Royal Pass.
- O No, compile error but it'll not print Royal Pass.
- O Compile-time error.

In every iteration the value of i is getting incremented by 1, (we can cancel the operation—i, with ++i, as these operations are there just for creating confusion). So initially the value of i=1, it will get incremented every time and print Royal Pass until the if condition gets false i.e; the value becomes 5, hence this code will print Royal Pass 4 times

Question 3 Time: 00:00:03

What will be the output of the following code?

```
#include<stdio.h>
int main ()
{
   int x = 4, y = 0;
   int z;
   z = (x++ +++y + y++ , x++);
   printf ("%d\n", z);
   return 0;
}
```



The code is free of errors in the line where

```
z = (x++++++y+y++, x++);
```

Here z will get the value of the last expression of this set.

from the first expression which is x+++++y+y++ the value of x will be 5.

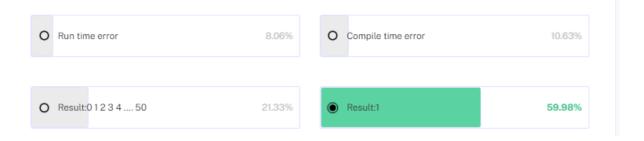
As, x++ is post increment hence z will get 5 first then x will be incremented. So, the answer will be 5.

Question 4

What will be the output of the following code?

```
#include <stdio.h>
int f(int n)
{
   if (n==0)
   return 1;
   else
   return f(n-1);
}

int main ()
{
   printf ("Result:%d", f (50));
   return 0;
}
```



The code is a basic application of recursion technique which prints a user required result, the function f(int n), will keep on calling itself until the value of n becomes 0, and then finally it will print 1

Question 5 Time: 00:00:01

What will be the output of the following C code?

```
#include <stdio.h>
int f(int n)
{
    if (n==0)
    return 1;
    else
    return n+f(n-1);
}
int main ()
{
    printf ("%d", f (10));
    return 0;
}
```

| O compile time error | O infinite loop |
|----------------------|-----------------|
| | |
| O 56 | O 55 |

The code is an example of recursion, where the function f(int n), call itself again and again until the value of n becomes zero, and on every call it is adding up the values of n, i.e; the function is adding 10+9+8+7+6+5+4+3+2+1+1=56, the extra 1 will be added when the value of n becomes zero, hence the output will be 56

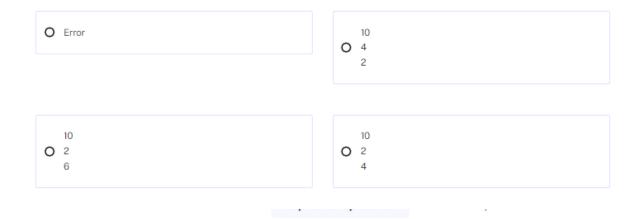
Question 6



What will be the output of the following C code?

```
#include <stdio.h>
int main ()
{
   int x = 5, y = 4, z = 3;
   int a = x<<+1;
   int b= y>>+1;
   int c = z<<+1;
   printf ("%d\n", a);
   printf ("%d\n", b);
   printf ("%d\n", c);

   return 0;
}</pre>
```



The following code is performing bitwise shift operations on the value of \boldsymbol{x} , \boldsymbol{y} and \boldsymbol{z}

```
So, a becomes x<<+1=5*2=10
```

b becomes y>>+1=4/2=2

c becomes z<<+1=3*2=6

Hence the answer is

10

2

6

Question 7 Time: 00:00:01

What will be the output of the following C code? [assume it is a 64 bit-computer]

```
#include <stdio.h>
struct node
{
    int data;
    int *pointer;
};
int main ()
{
    node m;
    printf ("%d", sizeof (m));
    return 0;
}
```

| O 4 | O 12 |
|------|------|
| | |
| O 16 | O 5 |

This happens due to Structural Padding. In a 64 bit computer, the frames take 8 bytes, and the variables are stored in frames. That's why if you take another integer variable after the first one, the total memory of the structure will remain the same.

In our case the data variable stays in a 8 bytes memory and pointer in stays in another 8 byte memory.

Hence 16 byte is the answer.

Question 8 Time: 00:00:00

What will be the output for the pseudocode for x=10, y=25

```
fun(int x,int y)

if(x=0)

return y;

else

return fun(x-1 , y+1)
```

```
O 25
O 9
O None of the above
```

```
int fun (int x, int y)
{
    if (x == 0)
    return y;
else
    return fun (x-1, y+1);
}
int main()
{
    int y=10;
    int x=25;
    printf("%d",fun(x,y));
}
```

The above code uses the recursion of a function to reach the final output. On every function call, the value of x is decreasing by 1 and the value of y is being updated as 'y+1'. So, in the final call the value of x=0 and y=35, this will be executed and 35 will be printed as the final output.

Question 9



What will be the output of the following code?

```
#include<stdio.h>
int main ()
{
   int no = 1112, temp, digit, sum = 1;
   temp = no;
   while (no > 0)
   {
      digit = no % 10;
      sum = sum * digit;
      no /= 10;
   }
   printf ("%d\n", sum);
   return 0;
}
```



The above code is doing the multiplication of digits of the given number, i.e., $1 \times 1 \times 1 \times 2 = 2$, which is the output of the given code.

Question 10 Time: 00:00:01

What will be the output of the following C code

```
#include<stdio.h>
int main ()
{
  int x = 21, y = 10, z = 13;
  x+y>5 ? (printf ("%d", z)) : (return z);
}
```

| O error | O 10 |
|---------|------|
| | |
| O 2 | O 13 |

expr1?expr2:expr3 works as follows: expr1 is evaluated first if it is true, expr2 is evaluated, otherwise expr3 is evaluated. Hence in expressions the **return** statement can not be used in **C**-language. The **return** statement is used for **returning** from a **function**, you can't **use** inside **ternary operator**

| a series in desire | Pseudocode-6 |
|--------------------|---|
| | |
| | Steps:- |
| | |
| |)initialisesx=0,y=1; |
| 1 | Tonloop Stoucture: |
| - ' ' | No initialization |
| | Condition: xisnon-zero |
| | Cont & + xx < x + + « « » |
| | |
| 11 | ++xis invalid, zhoreasenovariablexx |
| Marie Contract | 10 F 100 - 0 1 . 1 . 1 . 1 . 1 |
| 1 | compilationessessingical estas |
| • | Hypothetical output (if ++ x whinetx) |
| | Duto 1:- C- + : - : |
| | Cutput: - Contains conpilation essandu to undefined variable |
| 3 | |
| 3 | i) initialization: - i - 1, 1 is uninitialization |
| September 1 | |
| | ii) infinite Loup(foo(si)). |
| | |
| | It suns Indefinitely undell Bacariserne |
| i | Transition of the state of the |
| | Delinet iteration (i=1). |
| | 1++->i-> |
| | 1 + (i) is tomes) is = -i => 1 = 1 i=1. |
| | |

Ödödááá Pars, 8 Montet of the busines and barring. in Secondilenation (1=2): 1++-11=3 18 Ci) is Tome Ci=3 1, So 1===1-21-21 if Cicolistome (225) Paint « Royaland Sitt+ Die 3. I Thind iteration (i=3): 1++-) 1=4. 1+ Ci)is 70m Ci=a), Ca, 1'=--1-31=3. pars : 8 1++ 21=1 pani 1 "Royal" vi) Fount Hesation (1-4): · 2=1 C++i 12 (i) is bounc (i=c), so 3=--i-)/-4 Sit Cic Sisteme (ACS) point "Poral paris Sit + + Dies Vill Filthitenation Class. 12 +-) 1= 6 in (Dictory Cies) 20:==-1-27 serve 37-10-10-1

doublett- «poyapase» printa Hussen 3)i) initialization: x = 4, y=0 ii) Experssion >= (x+++++x+++++x+ iii) x++: x freturns 4, then x-5

+xx: Should be +xox+x

+x* y =) +4*0=0

+x+y,=)+4+0=4 y++: · y ord word or then y =1 Ans: 2-5+1-64 Output! - code contain comprehenses A)i) f (int n):
if n=-0, ordum; (Base com) Petuneire coul: P(n-1) until n-20. il) Call & (80) if no turnively Called & (49). - \$(48)... \$(0) Lubonn--0, fra Jahunni, does hexist is) Code will not corpile ductolinde lind, vanished. Compiletine ensor).

Stepi-DECIMEN! Cniticalennon: co-pilationennon - Recursive case: nx fm-1). Criticalensor: l'istadefine d'here: (B) Caus + (10), 10 x + (10-i) 3 possible: - 1-1, factorial of 10 1-2, product of alt nos explose of sub pilanci siopos prigires. tion of1. = intaata: size - Abytes int*pointer: size - 8 bytes · Lotal Size of Stand rodem = sized (int data) + Size of Clot * pointer) - Ati Paint (12) Cpaint & Cor o Jodon, Rimalio 6 Cleps:-M. Intholization! X=2, Y=1, S=3 2-Bituise operations:

0 × 12 - 1 16 left snift sbylbit: 5 Inbinarije let -) ghillestellbylgives Ino -10 D=y>>1: Pightshift 4 by Lunit: Ainbiner is 100 -) Chift 2 1ght by 1 give 10 binary deciral: C=>>>+ili 1 ell Shift 3 by 1 bit 1 3 inbinary is 11 Shift 1 elph by 1 give 110 1 binary 1 - 6 caucital; ・カーメンシャレジ · C=>26+1; (8) tired point (6) i) fun(25,10) -) calls for (24,10) Pun(24,11) -) Call fum(23,12) (un(1,34)) call tun(0,35) Can (0,35)) return 35 (base case) ! Pinaly - initialy + initial x Initial x-25 12 7=10 Final swell = 10+25=35, 9/i)'initial yolus: no= 1112, Sun=1 Di) Finet iterations - digita 1112 1010=2 Sum = 182-2

COMPASS M T W T F S S Date: mo-1112/10-11 · Precondition . digit - 111 *1010-Qum - 2 +1-2 mo-111/10=(1) Thirditeration! aigit=11'1010= Bum = 2 A/ = 2 mo=11/10=1 -digital °1010= Founttitenation: Sum = 2 x1-2 No= 1/10-10string morgang-ituation