WEEK 4 ASSIGNMENT SOLUTION

1. The loop which is executed at least once is
   a) while
   b) do-while
   c) for
   d) none of the above

Solution: (b) do-while loop is executed at least one even though the condition is false.

2. What is the purpose of the given program? n is the input number given by the user.

```
#include <stdio.h>
int main()
{
   int n, x = 0, y;
   printf("Enter an integer: ");
   scanf("%d", &n);
   while (n != 0)
        {
      y = n % 10;
      x = x * 10 + y;
      n = n/10;
   }
   printf("Output is = %d", x);
   return 0;
}
```

   a) Sum of the digits of a number
   b) Sum of the factorial of individual digits in a number
   c) Reverse of a number
   d) The same number is printed

Solution: (c) Reverse of a number. Please take a number and follow the operation step-by-step. You will be able to find the reversed number as output.

3. What is the output of the following C code?

```
#include <stdio.h>
int main()
{
   int a = 1;
   if (a--)
      printf("True\n");
   if (++a)
      printf("False\n");
return 0;
}
```

a) True
b) False
c) Both ‘True’ and ‘False’ will be printed
d) Compilation error

Solution: (c) ‘a--’ post-increment the value of a. Thus, the if statement is executed as the value of a is considered as 1 which is true. ‘++a’ pre-increment the value of a. Thus, the decremented value of a (which is 0) is incremented first and then assigned. So, both the if statements are executed ad correspondingly both True and False will be printed.

4. What will be printed when the following C code is executed?

```
#include<stdio.h>
int main()
{
   if('A'<'a')
   printf("NPTEL");
   else
   printf("PROGRAMMING");
   return 0;
}
```

a) NPTEL
b) PROGRAMMING
c) No output
d) Compilation error as A and a are not declared as character variable

Solution: (a)
ASCII value of ‘A’ is 65, which is lesser than the ASCII value of ‘a’ (i.e., 97). Therefore, if condition is true and NPTEL is printed.

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

```
#include<stdio.h>
int main()
{
   int p,t,si;
   float r;
   p=5000; t=4; r=7.5;
   si=(p*t*r)/100;
   printf("%f",si);
   return 0;
}
```

a) 1500.000000
b) 0.000000

c) 1400.000000
d) Compilation error

Solution: (b) si is declared as integer variable, but printed as float. So it will print 0.000000

6. In the following example, tell which statement is correct

```
if( (condition1==1) && (condition2)==1))
    printf("SUCESS");
```

a) Condition1 will be evaluated first, condition2 will be evaluated second
b) Condition2 will be evaluated first, condition1 will be evaluated second
c) Condition1 will be evaluated first, condition2 will be evaluated only if condition1 is TRUE
d) Condition2 will be evaluated first, condition1 will be evaluated only if condition2 is TRUE

Solution: (c) Condition1 will be evaluated first; condition2 will be evaluated only if condition1 is TRUE. This is called the Short-circuited evaluation of the operators

7. What the following program will print?

```
#include<stdio.h>
int main()
{
int a=0101;
printf("\n a=%d", a);
return 0;
}
```

a) a=101
b) a=5
c) a=65
d) Compilation error: Not a valid number

Solution: (c)
If a digit is starts with 0, then the number is treated as Octal number. Here, a=0101 is the octal representation of 65.

8. Choose the correct output of the following C code.

```
#include<stdio.h>
int main()
{
   int var1=10, var2=6;
   if(var1=5)
   {
```

```
        var2++;
        }
        printf("%d %d", var1, var2++);
        return 0;
}
```

a) 10 6
b) 10 8
c) 5 7
d) 5 8

Solution: (c)

Firstly, var1=5 is an assignment operation that results always TRUE. Therefore, the if condition is executed.
Secondly, var2++ increases the value of var2 by one. i.e., var2=7.
Finally, inside the printf statement, var1=5 and var2++=7 will be printed. Remember, var2++ is a post-increment operation, so for the printf statement, its value is not incremented to 8.

9. while(1) is used in a program to create

a) False statement
b) Infinite loop
c) Terminating the loop
d) Never executed loop

Solution: (b) while(1) is used to create infinite loop.

10. What will be the output? (&& is logical AND operation)

```
#include <stdio.h>
int main()
{
   int i=0, j=1;
   printf("\n %d", i++ && ++j);
   printf("\n %d %d", i, j);
   return 0;
}
```

a) 0
   1 2
b) 1

1 1

c) 0
0 0

d) 0
1 1

Solution: (d)
Inside the first printf statement, i++ results in 0 (due to post increment operation). As the left operand of && operator is zero, the compiler will not evaluate the right operand (i.e., ++j). So, j is not incremented after the first printf statement. Finally, the result is i=1 and j=1.