Theory assignment 1

1.

```
#include <stdio.h>


int main()
{
   static int i;
   for(i++;++i;i++)
   {
      printf("%d",i);
      if(i==6)
      break;
   }


   return 0;
}
/*
output = 246
```

Static int it initializes value and declared as 0 and the loop runs, first it goes in to initialization step and then condition and into the body of loop and then to the iteration and again checking condition it breaks when the condition for break is met */

2. Point out the difference between the constants 7, '7', and "7"

Ans: The difference between 7, '7', and "7"is 7 is an integer and '7' is a single character 7 and “7” is a string that contains a single character 7.

3.

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

```
int main()
{
unsigned char a=50;
char loop;
for(loop=7; loop>=0; loop--)
{
printf("%d ",(a & (1<<loop))?1:0);
}
return 0;
}
```

/*

Output=00110010

Binary of 50 is 110010 and the left shift is $1 \times 2^7 = 128$

128 in binary is 10000000

And then for ternary operator 0 is printed for false and 1 for true

And in the next iterations the loop values changes to 6,5,4,3,2 and finally 1.

And by solving every case we get 00110010

4. Mention the use of the register storage class and explain why the ampersand operator is not allowed on register variables

Ans: Register storage class is used to allocate memory in the CPU so that the compiler can access the variable can be accessed easily and fast as it is stored in CPU register it cannot be dereferenced so we cannot use & operator.

5. For which reason pointers are used to pass arguments in the call by reference method for function

Ans: In call by reference method, we need to pass the actual address of the variables so, we use pointers for passing the address of the variables.

6. #include <stdio.h>

int main()

{

```
while(1)
{
printf("Hello");
}
return 0;
}
```

Ans: As the while has 1 as a condition it means that the while loop is always true, so it is an infinite loop, so it prints “hello” continuously.

PART-B

7.

i. a=10+2*12/ (5%3) +5

The value of a is= 27. By using operator precedence first, the bracket executes first 5%3=2 a next division executes and then multiplication takes place and finally addition takes place

ii.
```
int main ()
{
   int a=0, b;
a=(5>2)?b=6:(b=8);
printf("%d%d",a,b);
return 0;
}
```

The output is 66 because 5>2 is true the ternary operator goes to b=6 and a value also becomes 6 so output is 66.

8

a)
```
#include <stdio.h>
int main ()
{
int i=0, j=0;
for (i=0; i<5; i++)
{
for (j=0; j<4; j++)
{
if(i>1)
break;
}
printf("Hi ");
}
}
```

Ans: the outer loop runs 5 times then the inner loop runs only 4 times the loop breaks when i>1 so it runs only 4 times. as the outer loop runs 5 times in the output Hi is printed 5 times.

b)
```
#include <stdio.h>
#define loop while (1)
```

```
int main ()
{
loop;
printf("Infinite");
return 0;

}
```

Ans: as the while (1) ends with a semicolon it does not print anything

9. Differentiate between break and continue keywords with examples?

ANS: if break statement is encountered in a loop then the execution exists out from the loop if we use continue then the skips the current iteration and continues from the next iteration.

PART-C

10. Write a C program to find the product of digits in a number. For example, if the input number is 21455, the output is 200. Don't use the concept of arrays

ANS:

```
#include <stdio.h>

int main()
{
   int a,r,p=1;
   printf("enter a number:");
   scanf("%d",&a);
   while(a!=0)
   {
      r=a%10;
      p=p*r;
      a=a/10;
   }
   printf("%d",p);

   return 0; }
```

12.1

a)

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

```
{

int a=0;

a=5>2? printf ("4"):3;

printf("%d",a);

return 0;

}

/* solution=41

here in the condition a=5>2 is true so the terinary operator prints 4 and later as a beecomes true a will become 1 and prints 1 */

b)  #include<stdio.h>
int main()
{
int a=0;
a=10+2*12/5*2+5;
printf("%d", a);
return 0;
}
/* solution=23
at first we take 'a' value as 0 later after performing the operation a will become 23*/


12.2

a)

#include <stdio.h>

int main()

{

int n;

for(n = 7; n!=0; n--)

printf("n = %d", n--);

return 0;

}
```

Here the output is an infinite loop as the condition is always true as it decreases from 7,5,3,1, -1 And so on…… so it is an infinite loop

b) #include<stdio.h>

int main()

{

```
static int var = 5;

printf("%d ",var--);

if(var)

main();

}
```

/* output= 5 4 3 2 1

here var is taken as 5 and it will perform var-- and decrease var value by 1 and the main recoures until var becomes 1 after becoming 1 if it becomes 0 and if (var) will not satisfies ass it's false nad it will stop printing */