Loop Statements

1. Objective

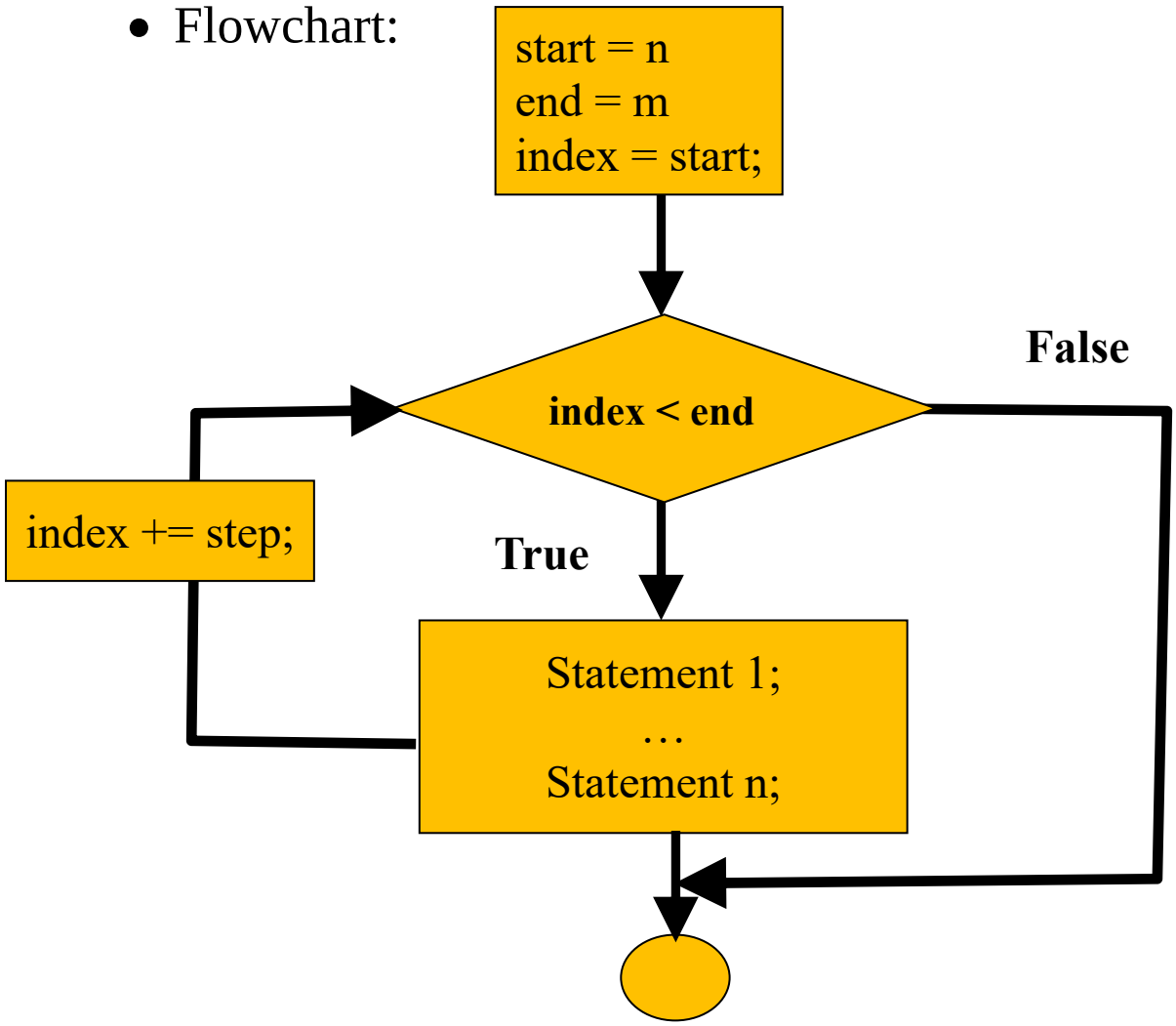
- Be able to repeat the execution of a set of statements a number of times.

- When one action is to be repeated a number of times a loop is used. Loops are ***repetition structures***

- There are three types of loop statements:
  - **for** loop
    - Specifically designed to repeat a set of statement a fixed number of times.
  - **while** loop
  - **do...while** loop
    - Used to continue repetition while a condition holds
    - Can also be used with a condition(s) to repeat a set of statement an unknown number of times.

2. Counting-Loop Statement

- **Use a Counting-Loop:** When you know how many time you want to execute a set of statements
- You would need to know how many iterations ahead of time
- The loop will run for each iteration unless there is a **break** in the body of your loop
- Flowchart:



- C implementation

  for (**variable initialization**; **condition**; **variable update**) {

  Statements

  }

- Infinite loop using **for** statement:
  - A loop that never ends:

```
//gcc 5.4.0
#include  <stdio.h>
#define maxNum   5
int main()
{
   int i=0;
   for(;;)
      printf("Hellooo\n");
}
```

- Examples:
  - List all the number less than 10

```
//gcc 5.4.0
#include  <stdio.h>
#define maxNum   10
```

```
int main()
{
   int i;
   for(i = 0; i<maxNum; ++i) {
      printf("i = %d\n", i);
   }
   printf("-------------------------------\n");
   printf("i after the for loop = %d\n", i);
   printf("-------------------------------\n");
   // for loop without the curly brackets {}
   for(i = 0; i<maxNum; ++i)
      printf("i = %d\n", i);
}
```

- Compute the sum

```
//gcc 5.4.0
#include  <stdio.h>

#define maxNum   5
int main()
{
   int i;
   int Sum = 0;
   for(i = 0; i<maxNum; ++i) {
      Sum += i;
  }
  printf("The sume is: %d\n", Sum);

}
```

- For loop using break

```
//gcc 5.4.0
#include <stdio.h>
#define maxNum   10
int main()
{
    int i;
    int Sum = 0;
    for(i = 0; i<maxNum; ++i) {
        if (i==1)
            break;
        Sum += i;
   }
   printf("The sume is: %d\n", Sum);
}
```

- Write a program that asks a use to enter 5 numbers and computes the sum of all entered numbers and their average.

```
//gcc 5.4.0
//Write a program that asks a use to enter 5 numbers and
computes the sum of all entered numbers and their
average.
#include  <stdio.h>

int main(void)
{
    int n;
```

```
    int counter=0;
    int sum=0;

    for(int i=1; i<=5; ++i)
    {
       scanf("%d", &n);
       if( n >= 0 ){
          ++counter;
          sum += n;
       }
       else
          break;

    }
    printf("The sum of all numbers is: %d\n", sum);
    printf("The average of all numbers is: %f\n", (float)
sum/ (float)counter);
}
```

- Compute the factorial of a a number. The factorial of n is: n! = n(n-1)(n-2)…1 and 0! = 1

```
//gcc 5.4.0

#include <stdio.h>

void main(void){

   int n;

   int f=1;

   int i;
```

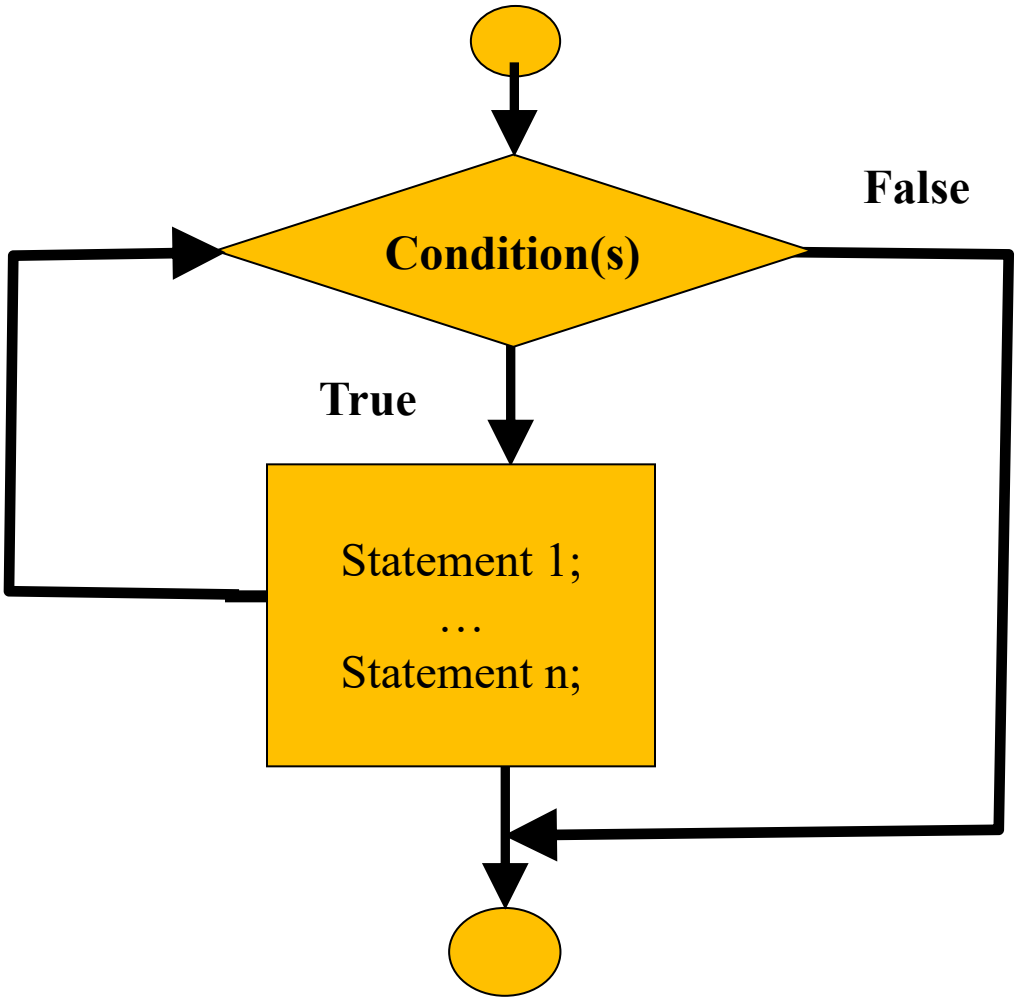
```
    scanf("%d", &n);
       printf("f= %d\n", f);
    if (n==0)
       printf("Factorial of %d!:%d\n", n , f);
    else if (n >0){
       for (i=1; i<=n; i++)
          f *= i;
       printf("Factorial of %d!:%d\n", n , f);
    }
    else
       printf("Factorial of %d! %s\n", n, "is not
possible");
}
```

- Write a program that asks a use to enter 5 numbers and computes their smallest number. Write a pseudocode for your code.

- Write a program that asks a use to enter 5 numbers and computes their largest number. Write a pseudocode for your code.

3. While Loop

- Use a While Loop: When to When you have no idea how many iterations you would need.
- Loop will stop executing when some condition becomes true or there is a **break** in the body of your loop.
- Flowchart:



- Ho While loop works?
  - The while loop evaluates the **condition**.
  - If the **condition** is true (nonzero), codes inside the body of while loop is evaluated.
  - The **condition** is evaluated again.
  - The process goes on until the **condition** is false.
  - When the **condition** is false, the while loop is terminated.
- C Implementation:

```
string ans = "n";

while   (Condition)
{
      Statements.
}
```

Can I put ; here?

No!!

Can I put ; here?

Up to you!!

- Infinite loop using While statement:

```
//gcc 5.4.0
#include  <stdio.h>
#define maxNum   5
int main()
{
   int i=0;
   while(1)
      printf("Hellooo\n");
}
```

- Examples:
  - List all the numbers less than 10 in descending order

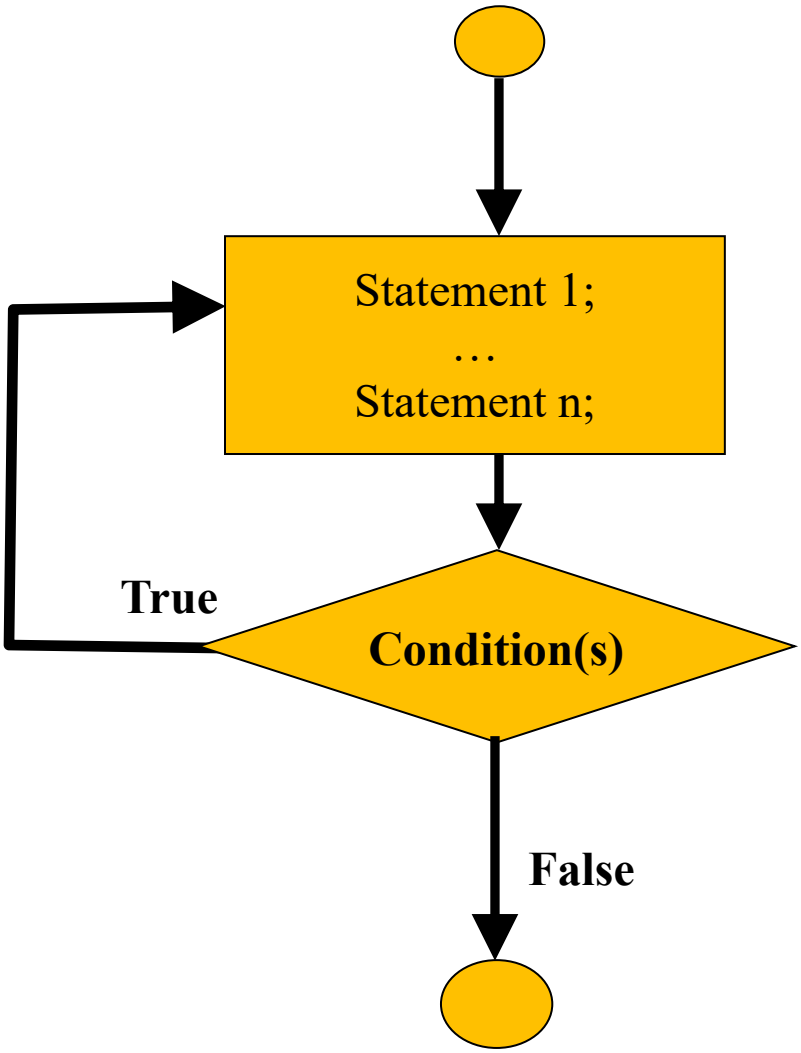
```
//gcc 5.4.0
#include  <stdio.h>
#define maxNum   10
int main()
{
   int i=10;
   int Sum = 0;
   while(--i){
```

```
        printf("i = %d\n", i);
        //++i;
    }
    printf("-------------------------------\n");
    printf("i after the for loop = %d\n", i);
}
```

- Rewrite the factorial program using while loop statement.
- Write a program that asks a use to enter 5 numbers and computes their smallest number. Write a pseudocode for your code. Use While statement.
- Write a program that asks a use to enter 5 numbers and computes their largest number. Write a pseudocode for your code. Use While statement.

4. do...while loop

- Cases where you have at least one iteration.
- **Flowchart:**



- **How do...while loop works?**
  - The **statements** inside the curly brackets are executed once.
  - Then, the **condition** is evaluated:
    - If it is **true**:
      - the loop body is executed again.
      - We start another iteration.
    - If the **condition** is evaluated to **false**, the do...while loop is terminated.

- **C Implementation**

```
do
{
      Statement;
} while (Condition);
```

; is required

- **Example:**
  - List all the number less than 5 and get their sum:

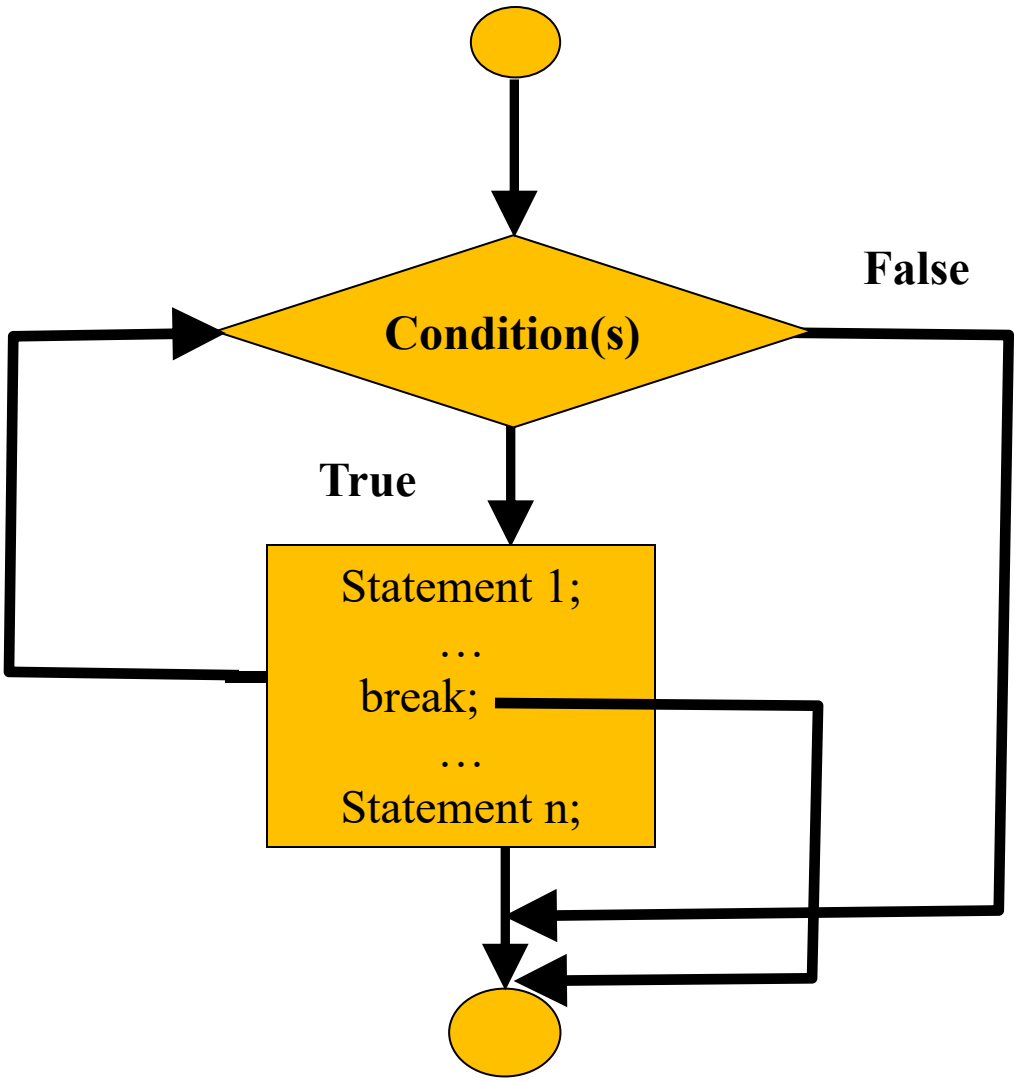
```
//gcc 5.4.0

#include  <stdio.h>

#define maxNum   5

int main()
{
   int i=0;
   int sum = 0;
   do {
     printf("value of i: %d\n", i);
     ++i;
   }while( i < maxNum );
   printf("--------------------------------\n");
   printf("i after the for loop = %d\n", i);
   printf("--------------------------------\n");

   i=0;
   do {
     sum += i;
     ++i;
   }while( i < maxNum );

   printf("The sum is: %d\n", sum);

}
```

5. The Break & Continue Statements: Loops

- Statements that causes you to exit the loop or continue to the next iteration.
- There are 2 special statements that can affect the execution of loop statements:
- Sometimes you would like to exit from a loop even if you still have more interactions to run.
- C Implementation:

```
break;
continue;
```

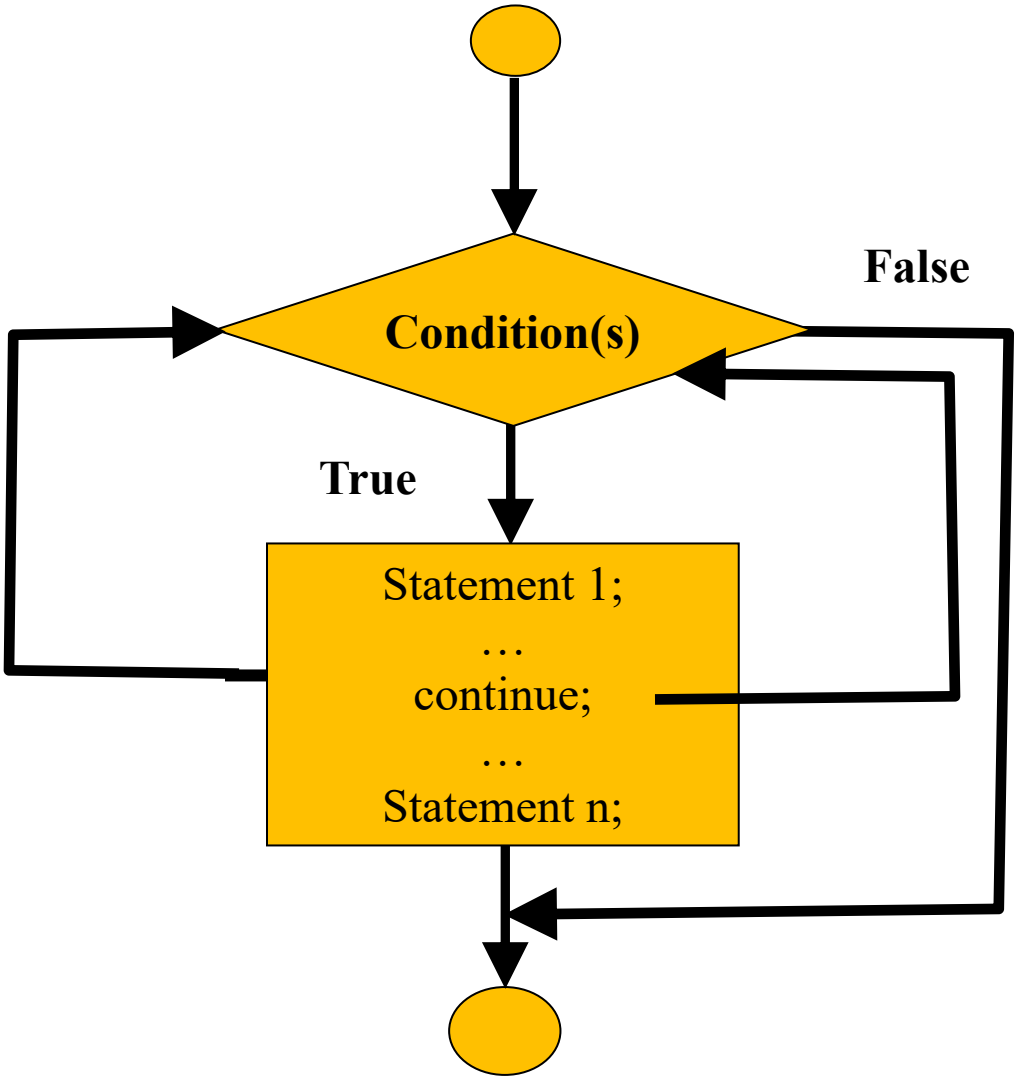
- Break statement:
  - When the break statement is executed inside a loop-statement, the loop-statement is terminated immediately.
  - The execution of the program will continue with the statement following the loop-statement.



- **Example:**
  - Write a program that asks a use to enter 5 numbers and computes the sum of all entered numbers.

```
//gcc 5.4.0
#include  <stdio.h>
int main(void)
{
   int n;
   int sum=0;

   for(int i=1; i<=5; ++i)
   {
      scanf("%d\n", &n);
      if(n < 0 )
        break;
      else
        sum += n;
   }
   printf("The sum of the entered numbers is:
%d\n", sum);
}
```

- Continue statement:
  - When the continue statement is executed inside a loop, the program will start another iteration.



- **Example:**
  - Write a program that asks a use to enter 5 numbers and compute all entered even numbers.

```
//gcc 5.4.0
#include  <stdio.h>
int main(void)
{
   int n;
   int sum=0;
   for(int i=1; i<=5; ++i)
   {
      scanf("%d\n", &n);
      if(n % 2 == 0 )
        sum += n;
      else
         continue;
   }
   printf("The sum of the even numbers is: %d\n", sum);
}
```

6. Nested Loops

- You can use a loop inside another loop since the
- **Examples:**
  - Generate the multiplication table of up to 10:

```
//gcc 5.4.0
//Write a program that generates the multiplication table up to 10
#include  <stdio.h>

int main(void)
{


   //Multiplication table:
   for(int i=0; i<=10; ++i){
      for (int j=0; j<=10; ++j)
         printf("%d--", i*j);
      printf("\n");
    }

}
```

  - What is the output of this code:

```
   //gcc 5.4.0

   #include  <stdio.h>
```

```
int main(void)
{
   int num, result, counter;
   printf("Enter a positive number:\n");
   scanf("%d", &num);
   result = 1;
   counter = 1;
   if (num != 0)
      do{
         result *= 2;
         counter++;
      }while (counter <= num);

   printf("Two raised to the %d power is:%d\n",
num, result);

}
```

- What is the output of this code?

```
//gcc 5.4.0

#include  <stdio.h>

#define MAX_ROWS 10

int main(void)
{
   for (int row = 1; row <= MAX_ROWS; row++)
```

```
    {
        for (int star = 1; star <= row; star++)
            printf("*");
        printf("\n");
    }
}
```

7. Scope of a Variable

- Scoping define the visibility of a variable in a list of statements, called block of code.
- Block of code in C is delimited by curly brackets
- Example:

- Example:

```
//gcc 5.4.0
//Write a program that asks a use to enter 5 numbers and computes the sum of all entered numbers and their average.
#include  <stdio.h>

int main(void)
{


   for(int i=1; i<=5; ++i)
   {
      {
         int j = 0;
         j += i;
         printf("j is: %d\n", j);
      }
      //printf("j is: %d\n", j);
   }

}
```

- Why the program output the following?

```
j is: 1
j is: 2
j is: 3
j is: 4
j is: 5
```

8. Questions/Practice

- What is the output of the following code?

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
   for(int i=0;;)
      printf("Hello, world!\n");
   return 0;
}
```

- What is the output of the following code?

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
   for(int i=0;i<=10;)
      printf("Hello, world!\n");
   return 0;
}
```

- Draw the flowchart of a for loop statement.

- What is the output of the following code?

```
//gcc 5.4.0


#include<stdio.h>
int main(){
   int num1,num2;


for(num1=0,num2=0;num1<=5,num2<=
4;num1++,++num2){
        printf("%d ",num1+num2);
   }

   return 0;
}
```

- Does the following code compile and if it does, what does it print?

```
#include <stdio.h>
   void main()
   {
      int k = 0;
      for (k)
         printf("Hello");
```

```
}
```

- Does the following code compile and if it does, what does it print?

```
#include <stdio.h>
 void main()
 {
    int k = 0;
    for (k < 3; k++)
    printf("Hello");
 }
```

- How many times “WOW” is get printed?

```
#include<stdio.h>
int main()
{
   int x;
   for(x=-1; x<=10; x++)
   {
      if(x < 5)
         continue;
      else
         break;
      printf("WOW");
   }
   return 0;
}.
```

- What is the output of the following code?

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
  for(int i =1; i<10; i++){
      for(int j = i+1; j<10; j++)
         printf("-");
      for(int k = 1; k<=i; k++)
         printf("%d", k);
      printf("\n");
   }
}
```

- Write a C program that prints the following:

```
1
22
333
4444
55555
…
nnn…….nnn
```

- What is the output of the following code?

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
   for(int i = 1; i<=10; ++i) {
      for (int k = 1; k<=i; ++k)
         printf("%d", k);

      for (int j = i; j>=1; --j)
         printf("%d", j);

      printf("\n");
   }
}
```