COMP133: **COMPUTER AND PROGRAMMING**

Iterative Logic

Dr. Radi Jarrar
Department of Computer Science
Birzeit University

Repetitions

- The repetitions in C is called loops.
- It is the process of repeating one or more commands based on some condition.
- As the condition remains true, the commands will be repeated. Once the condition is false, the loop will exit.

Repetitions

- There are 3 types of loops in C:

1. `while loop`
2. `for loop`
3. `do...while loop`

Controlling Loop Execution

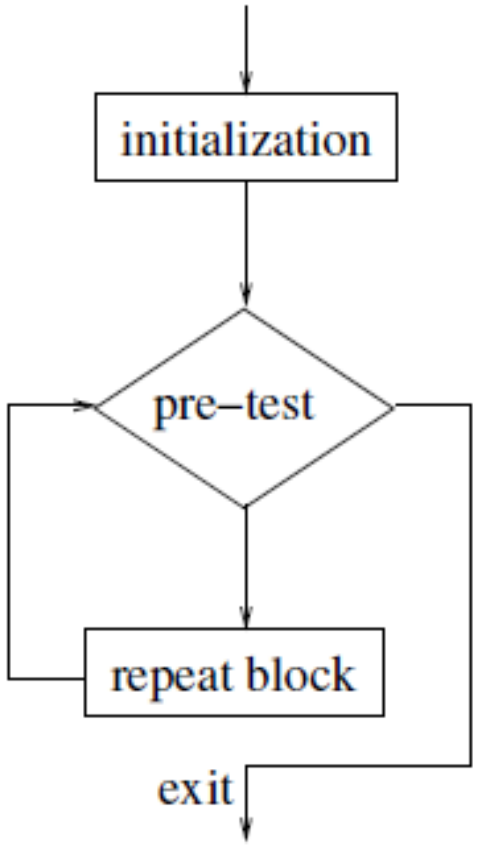
- Three methods:
  - **Counter controlled loops**: the execution of the loop is based on counter.
  - **Event controlled loops**: the loop continues until special value is encountered. (E.g., terminate loop when an input value of 'q' or -99 is entered).
  - **Result controlled loops**: the loop continues until a test determines that the desired result is reached (e.g., numerical approximations).

While Loop

- The same structure as seen previously in Algorithms.

```
Initialisation

while( condition )
{
  Statements
  Update
}
```



While Loop

- In-order to make sure a loop works successfully, you should maintain three main steps:
- **Initialisation (*start from*)** – the initial value of the loop control variable. Before the loop starts iteration.
- **Test (*stop at*)** – the loop condition will test the loop-control-variable before entering each loop iteration; if condition is true, the loop body is executed.
- **Update (*step*)** – loop control variable is updated in each iteration, typically at the end of the iteration.

Example

- Write a program to print the first 100 positive integers.

Example

```
#include<stdio.h>
int main(){
    int counter = 1;

    while( counter <= 100){
        printf("%d\n", counter);
        counter = counter + 1;
    }

    return 0;
}
```

Example

- Write a program that reads 20 grades and compute their average.

Example

```
#include<stdio.h>
int main(){
      int counter = 0, grade, total = 0;
      float average;

      while( counter < 20 ){
            printf("Please enter a grade");
            scanf("%d", &grade);

            total = total + grade;
            counter = counter + 1;
      }
      average = total / counter;
      printf("The average is %f\n", average);
      return 0;
}
```

Example

- Write a program that reads n grades and compute their average. When -1 is entered, stop.

Example

```
#include<stdio.h>
int main(){
	int counter = 0, grade, total = 0;
	float average;

	printf(“Please enter a grade”);
	scanf(“%d”, &grade);

	while( grade != -1){
		total = total + grade;
		counter = counter + 1;

		printf(“Please enter a grade”);
		scanf(“%d”, &grade);

	}
	average = total / counter;
	printf(“The average is %f\n”, average);
	return 0;
}
```

Example

- Write a program that reads n grades and compute their average. The number of students n is entered by the user.

Example

```
#include<stdio.h>
int main(){
        int counter = 0, n, grade, total = 0;
        float average;

        printf("Please enter the number of students");
        scanf("%d", &n);

        while( counter < n ){
                printf("Please enter a grade");
                scanf("%d", &grade);

                total = total + grade;
                counter = counter + 1;
        }
        average = total / counter;
        printf("The average is %f\n", average);
        return 0;
}
```

Example - Event controlled While

- Write a program to calculate the sum of a set of values (we don't know their count). When 0 is entered this means that program should stop receiving data, and print the sum.

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example

```
#include<stdio.h>
int main(){
	int sum = 0, x;

	printf("Please enter a value or 0 to stop");
	scanf("%d", &x);

	while( x != 0){		//when zero is entered, stop the program
		sum = sum + x;

		printf("Please enter a value or 0 to stop");
		scanf("%d", &x);
	}
	if( sum ) //or if( sum != 0 )
		printf("The sum is %d ", sum);
	else
		printf("Zero! No values were entered");
	return 0;
}
```


From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example - Result controlled while

- Write a program to calculate the sum of a set of values (we don't know their count). When the sum exceeds 1000 this means that program should stop receiving data, and print the number of values were entered.

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example

```
#include<stdio.h>
int main(){
      int sum = 0, counter = 0, x;

      while( sum <= 1000 ){   //exit when the sum exceeds 1000
            printf("Please enter a value ");
            scanf("%d", &x);

            sum = sum + x;
            counter = counter + 1;

      }

      printf("The sum is %d and their count is %d\n", sum, counter);

      return 0;
}
```


From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example

- Write a program to print the number of passes and the number of failures in a set of n students. The user should enter -1 to stop.

Example

```
#include<stdio.h>
int main(){
	int countPasses = 0, countFails;
	int x;

	printf("Please enter a value or -1 to stop");
	scanf("%d", &x);

	while( x != -1){    //when -1is entered, stop the program
		if( x >= 60)
			countPasses = countPasses + 1;
		else
			countFails = countFails + 1;
		printf("Please enter a value or -1 to stop");
		scanf("%d", &x);
	}
	printf("Number of passes is %d and number of failures is %d",
			countPasses, countFails);
	return 0;
```

Example

- Write a program to compute the factorial of a given number n.

Example

```
#include<stdio.h>
int main(){
      int factorial = 1, counter = 1, x;

      printf("Please enter a number");
      scanf("%d", &x);

      while( counter <= x ){
            factorial = factorial * counter;

            counter = counter + 1;
      }
      printf("The factorial of %d is %d", x, factorial);
      return 0;
}
```

Example

```
#include<stdio.h>
int main(){
     int factorial = 1, counter, x;

     printf(“Please enter a number”);
     scanf(“%d”, &x);

     for( counter = 1; counter <= x; counter++ ){
          factorial = factorial * counter;
     }
     printf(“The factorial of %d is %d”, x, factorial);
     return 0;
}
```

Example

- Write a program to check if an input number is prime or not.

Example

```
#include<stdio.h>
int main(){
        int isPrime = 1, counter = 2, x;

        printf("Please enter a number");
        scanf("%d", &x);

        while( counter < x ){   //when -1is entered, stop the program
                if( x % counter == 0)
                        isPrime = 0;

                counter++;
        }

        if( isPrime == 1 )
                printf("The number %d is a prime number\n", x);
        else
                printf("The number %d is NOT a prime number\n", x);

        return 0;
}
```

Pre/Post Increment

int j = 0;
j++; // Post-increment this will increment the value of j by 1
++j; // Pre-increment this will increment the value of j by 1

- Ex. `printf("%d\n", j++);`
- `printf("%d\n", ++j);`

- The same can be done for decrement:
- `int d = 6;`
- `printf("%d\n", d--);`
- `printf("%d\n", --d);`

Pre/Post Increment

Example:

```
int x = 2, y = 5;

int z = ++x * y --;

printf("%d\n", --z);
printf("%d\n", x++);
```

Compound assignment

```
int x = 10;
x += 2;
x -= 3;
x *= 5;
x /= 2;
```

Examples

```
Ex. int a=4, b=3, c=20;
c /= ++a;
printf("%d, %d, %d", a, b, c);
```

- Ex. int a=2, b=3, c=4;

```
c *= ++a * b++;
printf("%d, %d, %d", a, b, c);
```

Examples

- Ex. int i= 1;

```
while (i< 5)
    printf("%d " , i++);
printf("\nLast value of i is %d" , i);
```

- Ex. int a=2, b=3, c=4;

```
c *= ++a * b++;
printf("%d, %d, %d", a, b, c);
```

For loop

- The for loop provides a compact form for counter-controlled loops.

```
for( initialization; stopping-condition; update)
     statements;
```

Which is equivalent to the while loop:

```
initialization;
while( stopping-condition ){
     statements;
     update;
}
```

For loop

- Example: Write a for loop to print the numbers from 1 to 1000

```
int i;

for( i=1; i<=1000; i++)
    printf("%d", i);
```

For loop

```
for (x = 1, y = 0; x <= 100; x++)
     y += x;
```

- The initialisation-expression may contain initialisation of more than one variable: x to 1 and y to zero.

do...while loop

- The do...while differs from the while loop as it tests the condition after executing the body of the loop.
- This means, if the condition is false, then the do...while will execute the body of the loops at least once.
- Syntax of the do...while loop:

```
do
      statement ;
while (expression);
```

do...while loop

- Example: what is the output of the following loop?

```
x = 1;
do{
     printf ("%d\t", x);
     x = x + 1;
}while( x <= 5 );
```

do...while loop

- Example: what is the output of the following loop?

```
x = 10;
do{
     printf ("%d\t", x);
     x = x + 1;
}while( x <= 5 );
```

do...while loop

- Example: what is the output of the following loop?

```
do {
    printf("Enter a letter (from A to G)");
    scanf("%c", &letter);
} while(letter < 'A' || letter > 'G');
```

Break Statement

- The break statement is used for early exit from a loop.
- A break statement takes the control out of the loop.
- When break is encountered inside any loop, control automatically passes to the first statement after the loop.

Break Statement

```
#include<stdio.h>
int main(){
      int x;
      for( x=1; x<=10; x++)
      {
            if( x == 5 )
                  break;

            printf("%d\t", x);
      }
      printf("\nBroke out of loop at x == %d\n", x);
      return 0;
}
```

Continue Statement

- The continue statement is used to skip the rest of the loop, re-evaluate the condition of the loop again and repeat the iteration.
- It takes the control to the beginning of the loop.

Continue Statement

```
#include<stdio.h>
int main(){
     int x;
     for( x=1; x<8; x++)
     {
          if( x == 5 )
               continue;

          printf("%d\t", x);
     }
     return 0;
}
```

Break/Continue – Examples

What is the output of the following code?

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

    i = 1;
    while ( i++ < 7 )
    {

        printf("Hello\n");
        if ( i == 3)
            break;
        printf("Hi\n");
    }
        printf("Bye\n");
    return 0;
}
```

Break/Continue – Examples

What is the output of the following code?

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

    i = 1;
    while ( i++ < 5 )
    {

        printf("%d\n",++i);
        if ( i == 3)
            break;
        printf("%d\n",i);
    }
        printf("%d\n",++i);
    return 0;
}
```

Break/Continue – Examples

What is the output of the following code?

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

 int x=0 ;

 while(x++<=10) {

 if (x%2) continue;

 printf("%d\n" , x);

 }
    return 0;
}
```

Nested Loops --

Nested loop refer to a loop inside another.

```
#include<stdio.h>
int main(){
      int i, j;

      for( i=0; i<3; i++){
            printf("Outer i=%d\n", i);

            for( j=0; j<2; j++)
                  printf("i = %d\t j = %d\n", i, j);
      }
      return 0;
}
```

Nested Loops --

Nested loop refer to a loop inside another.

```
#include<stdio.h>
int main(){
     int i, j;

     for( i=0; i < 4; i++){

          for( j=0; j < i; j++)
               printf("i = %d\t j = %d\n", i, j);
     }
     return 0;
}
```

Nested Loops

Nested loop refer to a loop inside another.

```
#include<stdio.h>
int main(){
      int i, j;

      for( i=0; i < 4; i++){

            for( j=0; j < i; j++)
                  printf("*");

            printf("\n");
      }
      return 0;
}
```

Nested Loops

Nested loop refer to a loop inside another.

```
#include<stdio.h>
int main(){
	int i, j, space, rows = 5;

	for( i=0; i < rows; i++){
		for( space = 1; space <= rows - i; space++ )
			printf(" ");

		for( j=0; j < i; j++)
			printf("*");

		printf("\n");
	}
	return 0;
}
```

Nested Loops

Nested loop refer to a loop inside another.

```
#include<stdio.h>
int main(){
      int i, j;

      for( i=5; i > 0; i--){

            for( j=i; j > 0; j--)
                  printf("*");

            printf("\n");
      }
      return 0;
}
```

Examples

E.g., Print the even numbers between 1-100.

Examples

E.g., Print the even numbers between 1-100.

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

      for( i=1; i <= 100; i++){

            if( i%2==0 )
                  printf("%d\n", i);
      }
      return 0;
}
```

Examples

E.g., Print the numbers that are divisible by 3 between 1-100.

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

      for( i=1; i <= 100; i++){

            if( i%3==0 )
                printf("%d\n", i);
      }
      return 0;
}
```

Examples

E.g., **Write a c program to find out sum of digit of given number**

```
#include<stdio.h>
int main(){
        int num, sum = 0;

        printf("Please enter a number\n");
        scanf("%d", &num);

        while(num > 0){
                sum += num%10;
                num /= 10;
        }

        printf("The sum is %d\n", sum);
        return 0;
}
```