

# Introduction to Programming

Labor 03

# Exercise

What will be the value of the a, b and c variables after the execution of the following code.

```
int a = 15, b = 15, c = 15;  
c = (a%2) + (a=!b); printf("a=%d b=%d c=%d\n", a, b, c);  
  
int a = 2, b = 5, c = 15;  
c = a < b ? ++a : b++; printf("a=%d b=%d c=%d\n", a, b, c);  
  
int a = 2, b = 15, c = 1;  
b=4/3*c*c; a=b!=a; printf("a=%d b=%d c=%d\n", a, b, c);
```

# Exercise

What will be the value of the a, b and c variables after the execution of the following code.

```
int a = 15, b = 15, c = 15;  
c = (a%2) + (a!=b); printf("a=%d b=%d c=%d\n", a, b, c);  
  
int a = 2, b = 5, c = 15;  
c = a > b ? ++a : b++; printf("a=%d b=%d c=%d\n", a, b, c);  
  
int a = 2, b = 15, c = 1;  
b=4/3*c*c; a=b!=a; printf("a=%d b=%d c=%d\n", a, b, c);
```

# Statements

## Empty statements

Syntax can be of two types:

```
;  
{ }
```

## Semantics:

It does not do anything, but we may need it for syntactic purposes.

# Statements

## Syntax:

expression;

## Semantics:

Execution of the expression.

## Examples:

- `printf("Hello World!\n");`
- `x = 2;`

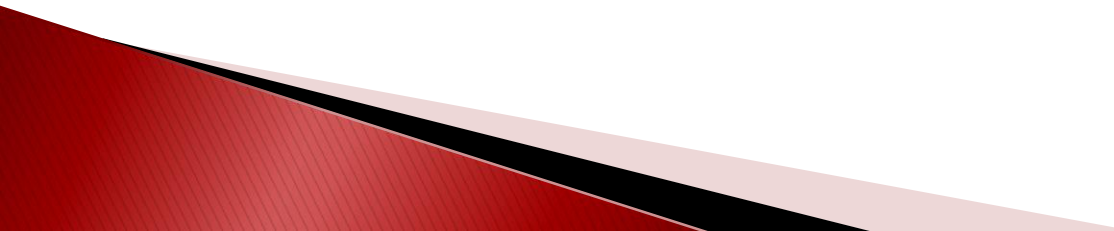
# IF statement

```
if (condition)  
    statement 1;
```

```
if (condition)  
{  
    statement 1;  
    statement 2;  
}
```

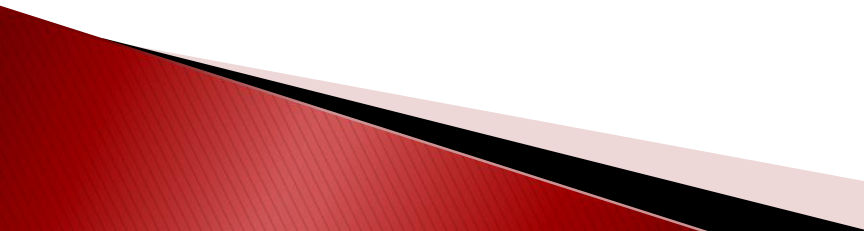
# IF-ELSE statement

```
▶ if (condition)
    statement 1;
else
{
    statement 2;
    statement 3;
}
```



# IF-ELSE-IF statement

```
if (condition)
    statement 1;
else if (condition)
    statement 2;
    .....
    .....
else if (condition)
    statement n-1;
else
    statement n;
```





# Example

```
if (x%2==0)
```

```
    printf("x is an even number\n");
```

```
else
```

```
    if (x>10)
```

```
        printf("x is an odd number and greater than 10\n");
```

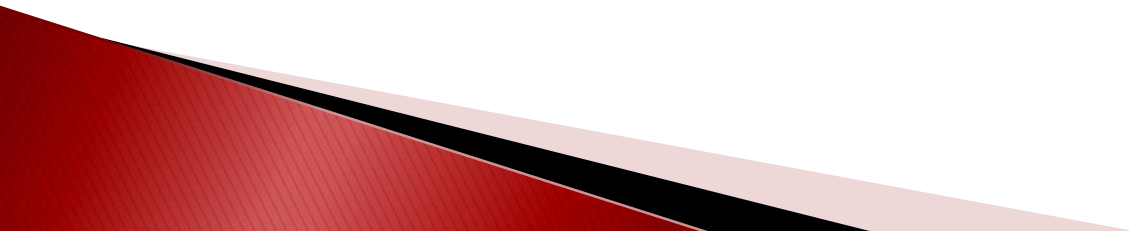
```
    else
```

```
        printf("x is an odd number and less than 10\n");
```



# Exercise

Write a program which determines if the input number is even or odd.



# Solution

```
#include <stdio.h>
int main()
{
    int a;
    printf("a=");
    scanf("%d", &a);

    if (a % 2 == 0)
        printf("%d is even\n", a);
    else
        printf("%d is odd\n", a);

    return 0;
}
```

```
gcc even.c -o even
./even
```

# Exercise

What will be the value of the  $k$  variable after the execution of the following code.

```
int i=7, j=5, k;  
if (i>=j && 0)  
    k=3;  
else  
    k=1;  
printf("k=%d\n", k);
```

# Exercise

What will be the value of the  $k$  variable after the execution of the following code.

```
int i=7, j=5, k;  
if (i>=j || 5)  
    k=8;  
else  
    k=5;  
printf("k=%d\n", k);
```

# Exercise

What is the result of this code?

```
int i=-5, j=3, k=1;  
if (i!=j || k)  
    k=-8 * (--i || j);  
else  
    k=-17 * (i++ && k);  
printf("i=%d\tj=%d\tk=%d\n", i, j, k);
```

# Exercise

What is the result of this code?

```
int i=5, j=10, k=2;  
if (j=k && i!=j)  
{  
    k+=j;  
    j+=--i;  
}  
else  
    k*=(i+j);  
printf("i=%d\tj=%d\tk=%d\n", i, j, k);
```

# Exercise

Write a program which determines if a triangle can be constructed from three segments.  
If it is possible, give the area of the triangle.



# Solution

```
#include <stdio.h>
#include <math.h>
int main()
{
    int a, b, c;
    float p, A;
    printf("a="); scanf("%d",&a);
    printf("b="); scanf("%d",&b);
    printf("c="); scanf("%d",&c);
    if (a<b+c && b<a+c && c<a+b)
    {
        printf("The triangle can be contructe!\n");
        p=(a+b+c)/2.0; //half perimeter
        A= sqrt(p*(p-a)*(p-b)*(p-c)); //triangle area
        printf("The area of the triangel is %.2f.", A);
    }
    else
        printf("The triangle cannot be contructe!\n");

    return 0;
}
```

```
gcc triangle.c -lm -o triangle
./triangle
```

# Exercise

Write a program, which evaluates a test on the basis of the obtained points.

## Example

point < 0 or point > 100 Default value!

point <= 20 Failed!

point <= 40 Grade is 2!

point <= 60 Grade is 3!

point <= 80 Grade is 4!

point <= 100 Grade is 5!

```
#include <stdio.h>
int main()
{
    int point;
    printf("point=");
    scanf("%d", &point);

    if (point < 0 || point > 100)
        printf("Default value!\n");
    else if (point <= 20)
        printf("Failed!\n");
    else if (point <= 40)
        printf("Grade is 2!\n");
    else if (point <= 60)
        printf("Grade is 3!\n");
    else if (point <= 80)
        printf("Grade is 4!\n");
    else
        printf("Grade is 5!\n");
    return 0;
}
```

# Solution

# Switch statement

► switch (expression)

{

case constant1: statements 1;

case constant2: statements 2; **break;**

.....

case constantn-1: statements n-1;

default: statements n;

}



# Exercise

Write a program, which qualifies the test on the basis of the obtained points. The points can be between 1 to 5, integer numbers [1,5].

## Example

point=1 Failed!

point=2 Grade is 2!

point=3 Grade is 3!

point=4 Grade is 4!

point=5 Grade is 5!

point<1 or point>5 Default value!

# Solution

```
#include <stdio.h>
int main()
{
    int point;
    printf("point=");
    scanf("%d",&point);

    switch (point)
    {
        case 1: printf("Failed!\n"); break;
        case 2: printf("Grade is 2!\n"); break;
        case 3: printf("Grade is 3!\n"); break;
        case 4: printf("Grade is 4!\n"); break;
        case 5: printf("Grade is 5!\n"); break;
        default: printf("Default value!");
    }
    return 0;
}
```

# Exercise

Write a program, which inputs an integer number between 1 to 5 [1,5] and prints '\*' characters that equals with the number.

- ▶ 1 \*
  - ▶ 2 \*\*
  - ▶ 3 \*\*\*
  - ▶ 4 \*\*\*\*
  - ▶ 5 \*\*\*\*\*
- 

# Solutions

```
#include <stdio.h>
int main()
{
    int stars;
    printf("stars=");
    scanf("%d",&stars);

    switch (stars)
    {
        case 1: printf("*\n"); break;
        case 2: printf("**\n"); break;
        case 3: printf("***\n"); break;
        case 4: printf("****\n"); break;
        case 5: printf("*****\n"); break;
        default: printf("Default value!");
    }
    return 0;
}
```

```
#include <stdio.h>
int main()
{
    int stars;
    printf("stars=");
    scanf("%d",&stars);

    switch (stars)
    {
        case 5: printf("");
        case 4: printf("");
        case 3: printf("");
        case 2: printf("");
        case 1: printf(""); break;
        default: printf("Default value!");
    }
    return 0;
}
```