

# Introduction to Informatics

Piroska Biró

# Type conversion

- ▶ Cast operator

Application: while evaluating the expression, a given variable should not be regarded according to the original type value.

**(type)** variable

Example:

```
float x=1.9;  
printf("%d", (int) x);
```

# Examples

- ▶ `int a=3, b=6;`  
`printf("%f", (float) a/b);`
- ▶ `float x=1.9;`  
`printf("%d", (int) (++x+3));`

# Exercise

What will be the value of the *dd* variable after the execution of the following code.

```
char q=5;  
int m=2;  
double dd, d=1.30;  
dd = (int)(q+m*d);  
printf("%lf\n",dd);
```

# Exercise

What is the result after the execution?

```
int ax, b;  
ax=0x97;  
b=0x24;  
ax= ax | b;  
printf("b=%x, ax=%d\n",b, ax);
```

# Exercise

What is the result after the execution?

```
int ax, b;  
ax=0x057;  
b=0x34;  
ax= ax & b;  
printf("b=%x, ax=%d\n",b, ax);
```

# Exercise

What is the result after the execution?

```
char ax, a, b=1;  
ax=0x057;  
a=0x033;  
ax |= (b << 3);  
a &= ~(b << 3);  
printf(" ax=%d\n  a=%d\n", ax, a);
```

# Exercise

What is the result after the execution?

```
char ax, a, b=1;  
ax=0x058;  
a=0x033;  
ax ^= (b << 3);  
a &= ~(b << 5);  
printf(" ax=%d\n  a=%d\n", ax, a);
```



# Exercise

What is the result after the execution?

```
double a=32.23, b=23.32, tmp;
```

```
tmp=a;
```

```
a=b;
```

```
b=tmp;
```

```
printf("\na= %lf \nb= %.3lf",a,b);
```



# Exercise

What is the result after the execution?

```
double a=32.23,b=23.32;
```

```
    a-=b;
```

```
    b+=a;
```

```
    a=b-a;
```

```
printf("\na= %.2lf \nb= %lf",a,b);
```



# Exercise

What is the result of this code?

```
int c1 = 12, c2 = 45;  
    c1 ^= c2;  
    c2 ^= c1;  
    c1 ^= c2;  
printf("c1 = %X\tc2 = %X", c1, c2);
```

# Exercise

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

```
int a, b, c;  
a = b = c = 11;  
c = ++a * (b%3);  
c = a < b ? a*4 : b/3;  
b--; a++; c--;
```

# Exercise

What will be the value of the i, j and k variables after the execution of the following code.

```
i=3; j=1; k=2;
```

```
k = i+++j;
```

```
k=++i+j++;
```

```
k=--i-j--;
```

```
k=-i+++j;
```

```
k+=++i+--j;
```

```
k+=-i+++j;
```

# Exercise

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

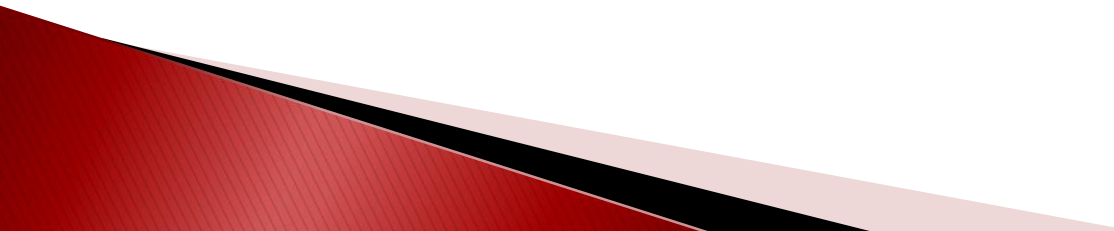
```
int i, j, k; i=7; j=5;  
if(i>=j && 0)  
    k=3;  
else  
    k=1;  
printf("%d\n",k);
```

# Exercise

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

```
int a, b, c;  
a = b = c = 9;  
c = a++ * (b%4);  
c = a < b ? a*2 : b/3;  
b--; a++;
```

# IF statement

- ▶ if (condition)  
    statement;
  - ▶ 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
    statemens n ;
```

# Example

```
if (x < 0)
    printf("The number is negative.\n");
else
    printf("The number is positive.\n");
```

```
if (x < 0)
    printf("The number is negative.\n");
else if (x == 0)
    printf("The number is zero.\n");
else
    printf("The number is positive.\n");
```

# Switch statements

- ▶ switch (expression)

```
{  
  case constant1: statements 1;  
  case constant2: statements 2; break;  
  .....  
  case constantn-1: statements n-1;  
  default: statements n;  
}
```

# Example

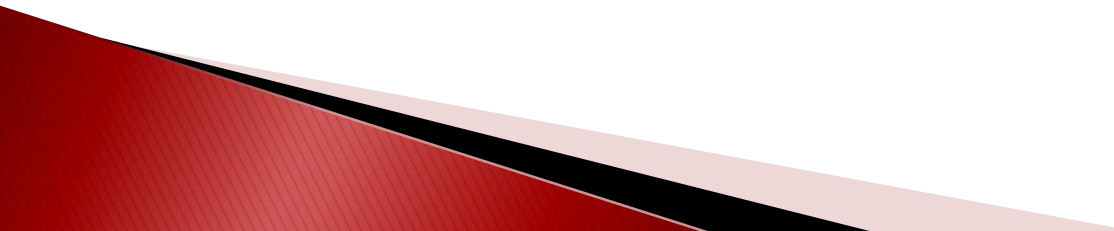
```
char op;  
int a, b, res;  
scanf("%d%c%d", &a, &op, &b);  
    switch (op) {  
        case '+': res = a + b; break;  
        case '-': res = a - b; break;  
        case '*': res = a * b; break;  
        case '/': if (b == 0)  
            printf("Error: Division by zero!");  
            else  
                res = a / b; break;  
        default:  
            printf("Default operation symbol!");  
    }  
    printf("Result:%d", res);
```

# FOR loop

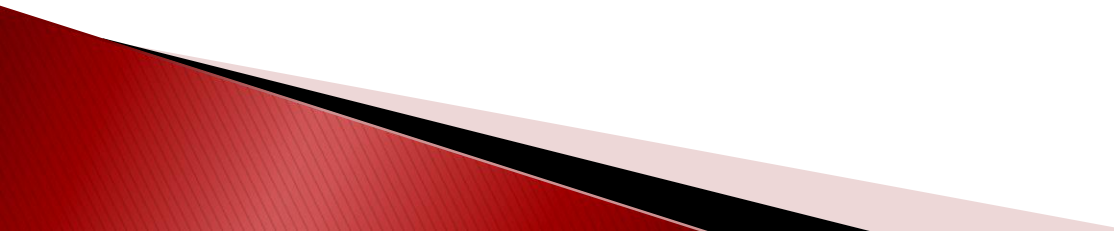
```
for (initialization_expression; loop_condition; increment_expression)
{
    statements;
}
```

## Example:

```
for(i=1; i<=n; i++)
{
    ...
}
```



# FOR loop

- ▶ Write a program which prints the first 10 integer numbers and their square.
  - ▶ Write a program which determines the **sum** and **product** of the first **n** number.
- 

# Solution

```
for (i=1;i<=10;i++)  
printf("%d squared: %d\n",i, i*i);
```



# Example

Sum of the numbers:

```
int i, n, sum;
```

```
sum=0;
```

```
for (i=1;i<=n;i++)
```

```
    sum+=i;
```

```
for (i=1, sum=0; i<=n; sum+=i++)
```

# Example

Product of the numbers:

```
int i, n, prod;  
prod=1;  
for (i=1;i<=n;i++)  
    prod*=i;
```

```
for (i=1, prod=1; i<=n; prod*=i++)
```

# Example

What is the result after the execution?

```
for (i=1;i<=10;i++)  
printf("i value: %d\n", i);
```

```
i=1;  
for( ;i<=10;i++)  
printf("i value: %d\n", i);
```

# Example

```
i = 1;  
for ( ; ; i++)  
{ if (i == 10) break;  
  printf("i value: %d\n", i);  
}
```

```
i = 1; for(;;)  
{ if (i == 10) break;  
  printf("i value: %d\n", i);  
  i = i + 1; }
```

# Exercise

What will be the value of the *j* variable after the execution of the following code.

```
int i, k=0, j=0;  
for (i=1;i<=4;i++)  
    j=k++;  
printf ("%d\n",j);
```

# WHILE loop

```
while (condition)
{
    statements;
}
```

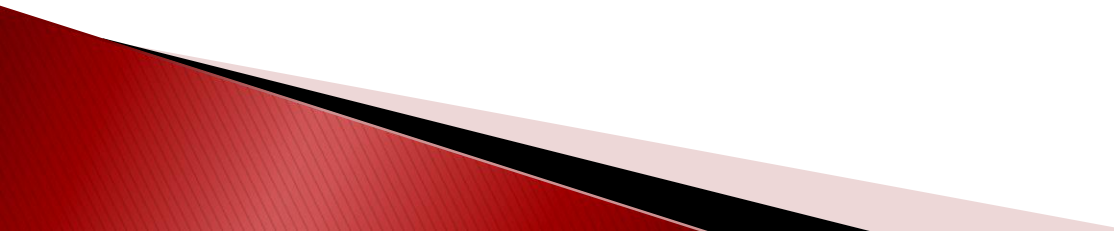
# Example

- ▶ Sum of the numbers:

```
i=0;  
while (i<=n)  
{  
    sum+=i;  
    i++;  
    /*sum+=i++;*/  
}
```

# WHILE loop – Exercise

Write a program which inputs the integer numbers from the keyboard until we type zero, and meanwhile it determines if the input number is even or odd.

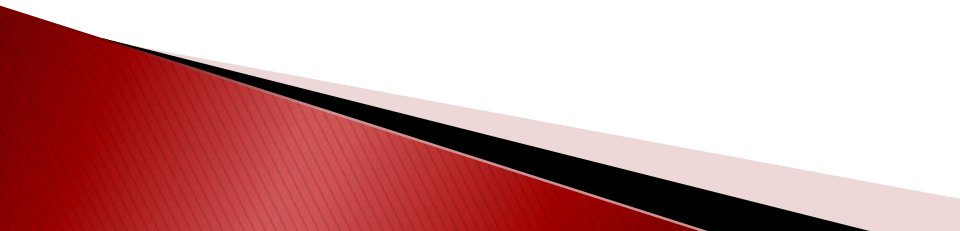




# Solution

```
printf("n=");  
scanf("%d",&n);
```

```
while (n!=0)  
{  
    if (n%2==0)  
        printf("%d even\n",n);  
    else  
        printf("%d odd\n",n);  
    scanf("%d",&n);  
}
```



# DO WHILE loop

```
do  
{  
    statements;  
  
} while (condition);
```

# Example

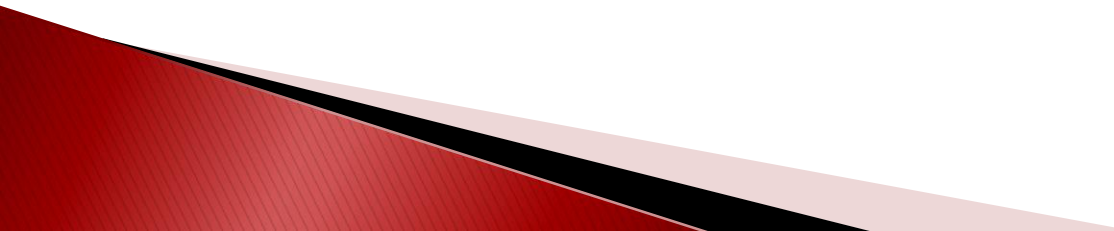
What is the result after the execution of this code?

Rewrite the code again using do while, and the result be the same.

```
int x, k = 0;
scanf("%d", &x);
while(x!=0)
{
    scanf("%d", &x);
    k++;
}
printf("%d", k);
```

# Solution

```
int x, k = 0;  
do  
{  
    scanf("%d", &x);  
    k++;  
} while(x!=0);  
  
printf("%d", k-1);
```



# Which are infinite?

1. `i = 1; while(!i) {...}`
2. `for (x = 1;; x++) {...}`
3. `i = 0; while(!i) {...}`
4. `int i=5; while(i == 10) i=10;`
5. `for(x = 10; x >= 10; x++) {...}`