Introduction to Programming in C

(IS-FEE-10061S)

Białystok University of Technology
Faculty of Electrical Engineering
Academic year 2023/2024

Workshop no. 03 (14.03.2024)

Jarosław Forenc, PhD

Topics

- The if statement
- Relational and logical operators
- The conditional operator: ? :
- The switch statement

Example: square root

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

int main(void)
{
    float x, y;

    printf("Enter the number: ");
    scanf("%f",&x);

    y = sqrt(x);

    printf("Square root: %f\n",y);

    return 0;
}
```

```
Enter the number: 15
Square root: 3.872983
```

```
Enter the number: -15
Square root: -1.#IND00
```

Example: square root

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

int main(void)
{
    float x, y;

    printf("Enter the number: ");
    scanf("%f",&x);

    if (x>=0)
    {
        y = sqrt(x);
        printf("Square root: %f\n",y);
    }
    else
        printf("Error! Negative number\n");

    return 0;
}
```

```
Enter the number: 15
Square root: 3.872983
```

```
Enter the number: -15
Error! Negative number
```

The if statement

```
if (expression)
    statement
```

- if expression is true (nonzero), statement is executed
- if expression is false (zero), statement is skipped (ignored)

```
if (expression)
    statement1
else
    statement2
```

- if expression is true, statement1 is executed and statement2 is skipped (ignored)
- if expression is false, statement1 is skipped (ignored) and statement2 is executed

- Expression in brackets:
  - true - when its value is different from zero (nonzero)
  - false - when its value is zero

The if statement

```
if (expression)
    statement
```

- Statement:
  - a single statement - one statement terminated by a semicolon
  - a compound statement (a single block) - one or more statements enclosed in braces

```
if (x>0)
    printf("stmnt1");
```

```
if (x>0)
{
    printf("stmnt1");
    printf("stmnt2");
    ...
}
```

The if statement

```
if (expr)
    stmnt;
```

```
if (expr)
{
    stmnt;
    stmnt;
}
```

```
if (expr)
    stmnt;
else
    stmnt;
```

```
if (expr)
{
    stmnt;
    stmnt;
}
else
{
    stmnt;
    stmnt;
}
```

```
if (expr)
{
    stmnt;
    stmnt;
}
else
    stmnt;
```

```
if (expr)
    stmnt;
else
{
    stmnt;
    stmnt;
}
```

```
if (expr)
{
    stmnt;
}
else
{
    stmnt;
}
```

Relational operatrors

Operator	Example	Meaning
`>`	`a > b`	a is greater than b
`<`	`a < b`	a is less than b
`>=`	`a >= b`	a is greater than or equal to b
`<=`	`a <= b`	a is less than or equal to b
`==`	`a == b`	a is equal to b
`!=`	`a != b`	a is not equal to b

- The result of the comparison is an int value equal to:
  - 1 - when the condition is true
  - 0 - when the condition is false (is not true)

Logical operators

Operator	Meaning	Description
!	NOT	!expr1 is true if expr1 is false, and it is false if expr1 is true
&&	AND	expr1 && expr2 is true only if both expr1 and expr2 are true
||	OR	expr1 || expr2 is true if either expr1 or expr2 is true or if both are true

- The result of the logical operators && and || is an int value equal to 1 (true) or 0 (false)

```
if (x>5 && x<8)
```

```
if (x<=5 || x>8)
```

Logical expressions

- Logical expressions may consist of:
  - relational operators
  - logical operators
  - arithmetic operators
  - assignment operators
  - variables
  - constant
  - function calls
  - …
- The order of operations depends on the precedence of the operators

Operator	Type
!	logical
* / %	arithmetic
+ -	arithmetic
> < >= <=	relational
== !=	relational
&&	logical
||	logical
=	assignment

(from high to low precedence)

Logical expressions

```
int  x = 0, y = 1, z = 2;
```

```
if ( x == 0 )
```

result: 1 (true)

```
if ( x = 0 )
```

result : 0 (false) (!!!)

```
if ( x != 0 )
```

result: 0 (false)

```
if ( x =! 0 )
```

result: 1 (true) (!!!)

```
if ( z > x + y )
```

```
if ( z > (x + y) )
```

result: 1 (true)

Logical expressions

```
int  x = 0, y = 1, z = 2;
```

```
if ( x>2 && x<5 )
```

```
if ( (x>2) && (x<5) )
```

result: 0 (false)

- Logical expressions are evaluated from left to right
- The calculation process ends when the result of the whole expression is known

```
if ( 2 < x < 5 )
```

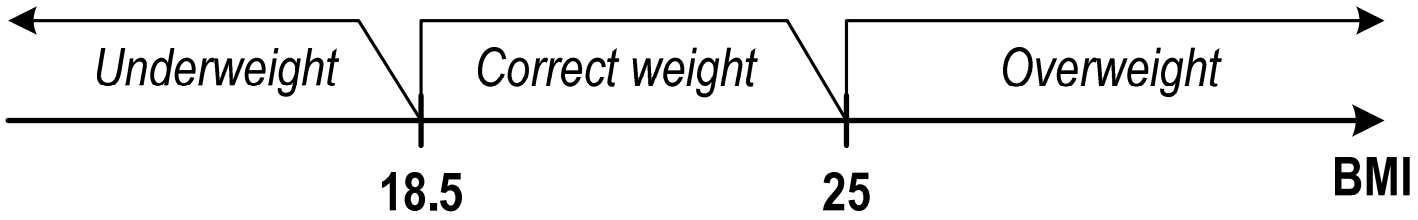
result: 1 (true) (!!!)

Example: BMI calculation (Body Mass Index)

- BMI - coefficient obtained by dividing body weight in kilograms by the square of height in meters

$$BMI = \frac{weight}{height^2}$$

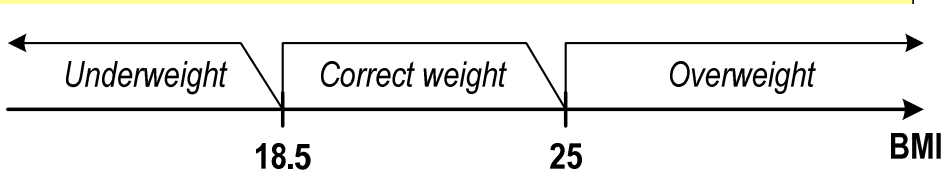
- For adults:
  - BMI < 18.5 - indicates underweight
  - BMI ≥ 18.5 and BMI < 25 - indicates the correct body weight
  - BMI ≥ 25 - indicates overweight



Example: BMI calculation (Body Mass Index)

```c
#include <stdio.h>

int main(void)
{
    double weight, height, bmi;

    printf("Enter weight [kg]:  "); scanf("%lf",&weight);
    printf("Enter height [m]: "); scanf("%lf",&height);
    bmi = weight / (height*height);
    printf("bmi: %.2f\n",bmi);

    if (bmi<18.5)
        printf("Underweight\n");
    if (bmi>=18.5 && bmi<25)
        printf("Correct weight\n");
    if (bmi>=25)
        printf("Overweight\n");

    return 0;
}
```

```
Enter weight [kg]:  84
Enter height [m]: 1.85
bmi: 24.54
Correct weight
```



The conditional operator

- The conditional operator consists of two symbols and three operands

```
expression1 ? expression2 : expression3
```

- Most often it replaces simple if-else statements

```
float duty, price, capacity;
```

```
if (capacity <= 2000)
    duty = price*0.031;    /*  3.1% */
else
    duty = price*0.186;    /* 18.6% */
```

```
duty = capacity <= 2000 ? price*0.031 : price*0.186;
```

The conditional operator

```
if (x < 0)
     y = -x;
 else
     y = x;
```

```
y = x < 0 ? -x : x;
```

- calculation of the modulus of x

```
if (a > b)
     max = a;
 else
     max = b;
```

```
max = a > b ? a : b;
```

- determining the max of two numbers

- The conditional operator has a very low precedence
- Only the assignment operators (=, +=, -=,...) and the comma operator (,) have lower precedence

Example: check whether the number is even/odd

```
#include <stdio.h>

int main(void)
{
   int x;

   printf("Enter x: ");
   scanf("%d",&x);

   if (x%2==0)
       printf("Even number\n");
   else
       printf("Odd number\n");

   printf("%s number\n",x%2==0 ? "Even":"Odd");

   return 0;
}
```

```
Enter x: -3
Odd number
Odd number
```

The switch statement

- The switch statement syntax

```
switch (expression)
{
    case constant1: statements
    case constant2: statements
    case constant3: statements
    ...
    default: statements
}
```

- constant - an integer value known at compile time
  - numeric constant, np. 3, 5, 9
  - character in single quotes, e.g. 'a', 'z', '+'
  - a constant defined by const or #define

The switch statement

- A program that prints in words a number from the range 1..5 entered on the keyboard

```
#include <stdio.h>

int main(void)
{
      int number;

      printf("Enter number (1..5): ");
      scanf("%d",&number);
```

The switch statement

```
switch (number)
{
   case 1: printf("Number: one\n");
           break;
   case 2: printf("Number: two\n");
           break;
   case 3: printf("Number: three\n");
           break;
   case 4: printf("Number: four\n");
           break;
   case 5: printf("Number: five\n");
           break;
   default: printf("Other number\n");
}
```

```
Enter number: 2
Number: two
```

```
Enter number: 0
Other number
```

The switch statement

```
switch (number)
{
    case 1:
    case 3:
    case 5: printf("Odd number");
            break;
    case 2:
    case 4: printf("Even number\n");
            break;
    default: printf("Other number");
}
```

```
Enter number: 2
Even number
```

- The same instructions can be executed for several case labels

The switch statement

```
switch (number)
{
   case 1: case 3: case 5:
           printf("Odd number\n");
           break;
   case 2: case 4:
           printf("Even number\n");
           break;
   default: printf("Other number\n");
}
```

```
Enter number: 2
Even number
```

- case labels can be written in one line

The switch statement

```
switch (number%2)
{
    case 1: case -1:
            printf("Odd number\n");
            break;
    case 0:
            printf("Even number\n");
}
```

```
Enter number: 2
Even number
```

- The default part can be omitted

The switch statement (without break)

```
switch (number)
{
   case 1: printf("Number: one\n");
   case 2: printf("Number: two\n");
   case 3: printf("Number: three\n");
   case 4: printf("Number: four\n");
   case 5: printf("Number: five\n");
   default: printf("Other number\n");
}
```

```
Enter number: 2
Number: two
Number: three
Number: four
Number: five
Other number
```

- Omitting the break statement will execute all statements after the given case (until the end of the switch)

End of workshop no. 03

Thank you for your attention!