Programming ||

Spring, 2023

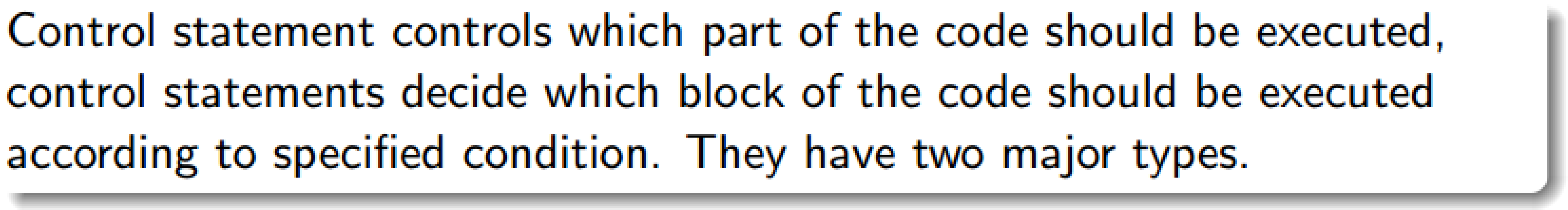
Musa Hodman

- Lecture 05
- Control Structure

Road Map

- Control statements
- Loops
  - The for loop
- The while loop
- The do Loop

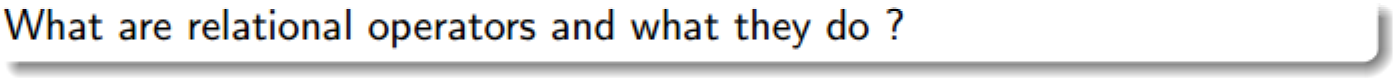
Control Statements



Control statement controls which part of the code should be executed, control statements decide which block of the code should be executed according to specified condition. They have two major types.

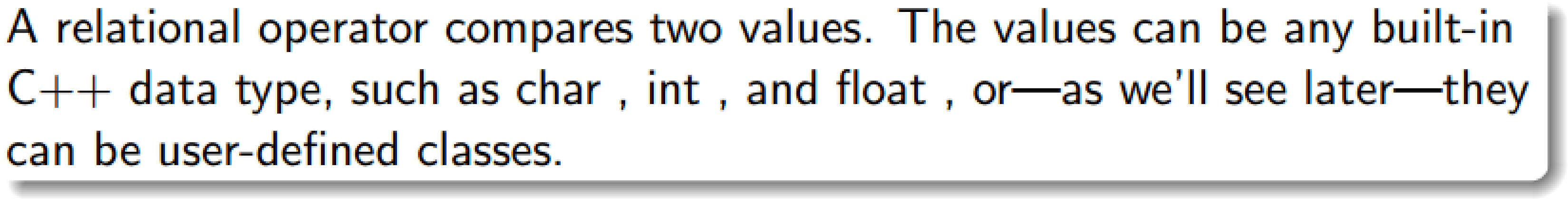
- Loops
- Decision

Question



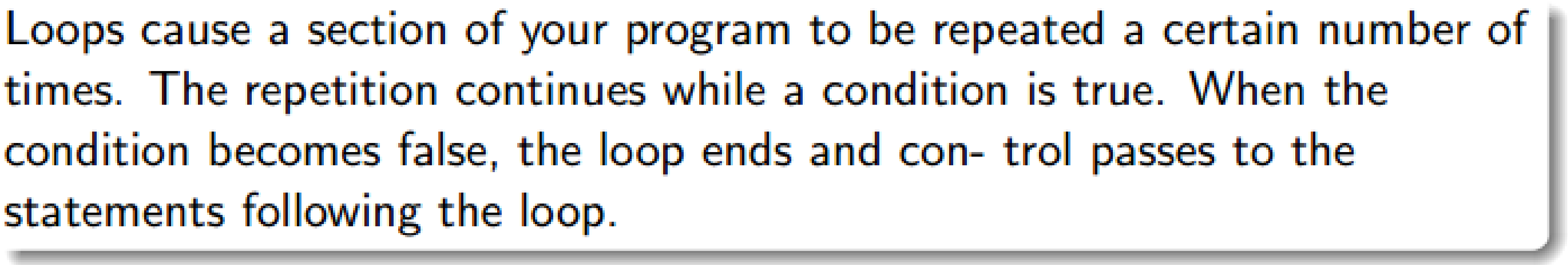
What are relational operators and what they do ?

Relational Operator

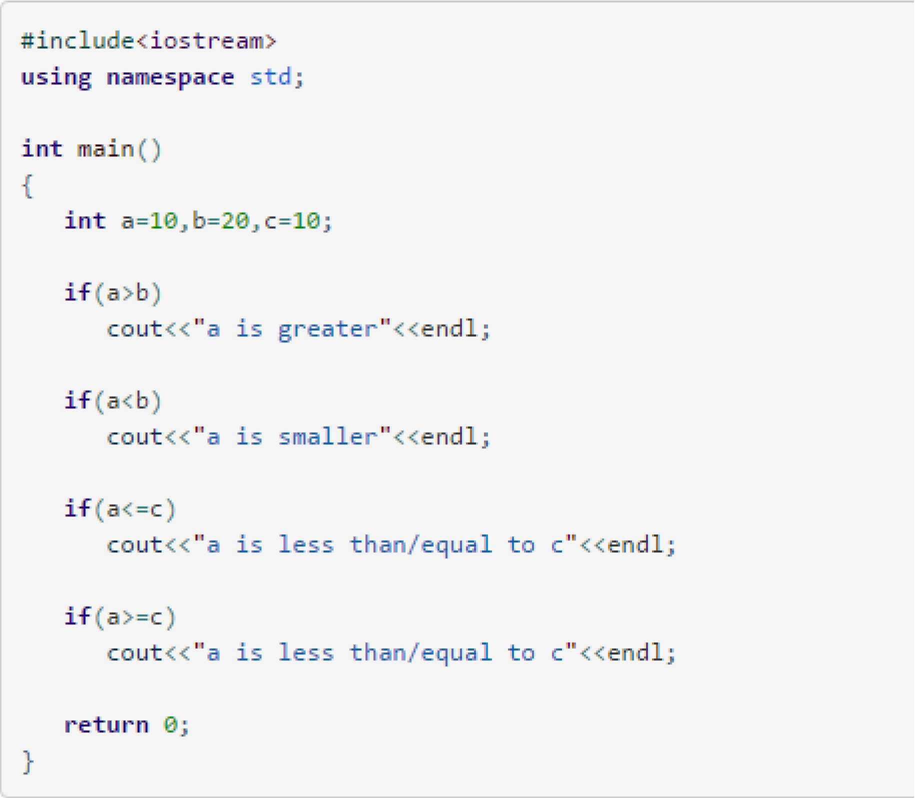


A relational operator compares two values. The values can be any built-in C++ data type, such as char , int , and float , or—as we'll see later—they can be user-defined classes.

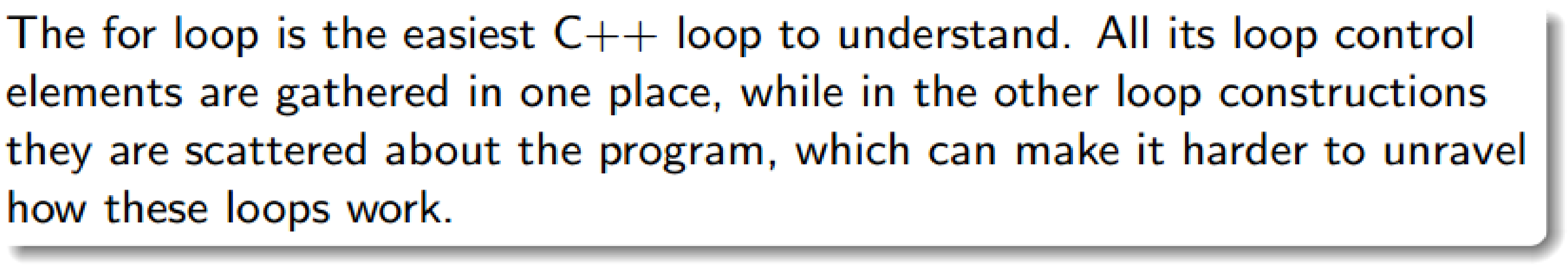
Loops



Loops cause a section of your program to be repeated a certain number of times. The repetition continues while a condition is true. When the condition becomes false, the loop ends and con- trol passes to the statements following the loop.

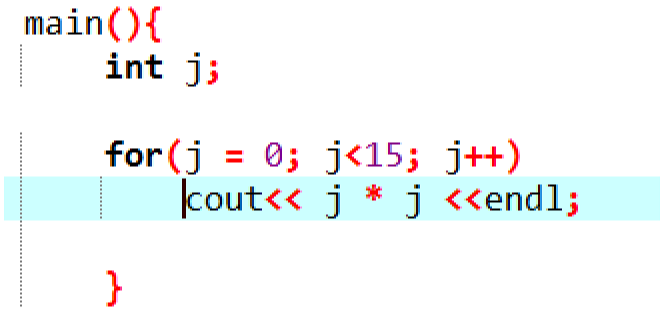


```
#include<iostream>
using namespace std;

int main()
{
   int a=10,b=20,c=10;

   if(a>b)
      cout<<"a is greater"<<endl;

   if(a<b)
      cout<<"a is smaller"<<endl;

   if(a<=c)
      cout<<"a is less than/equal to c"<<endl;

   if(a>=c)
      cout<<"a is less than/equal to c"<<endl;

   return 0;
}
```



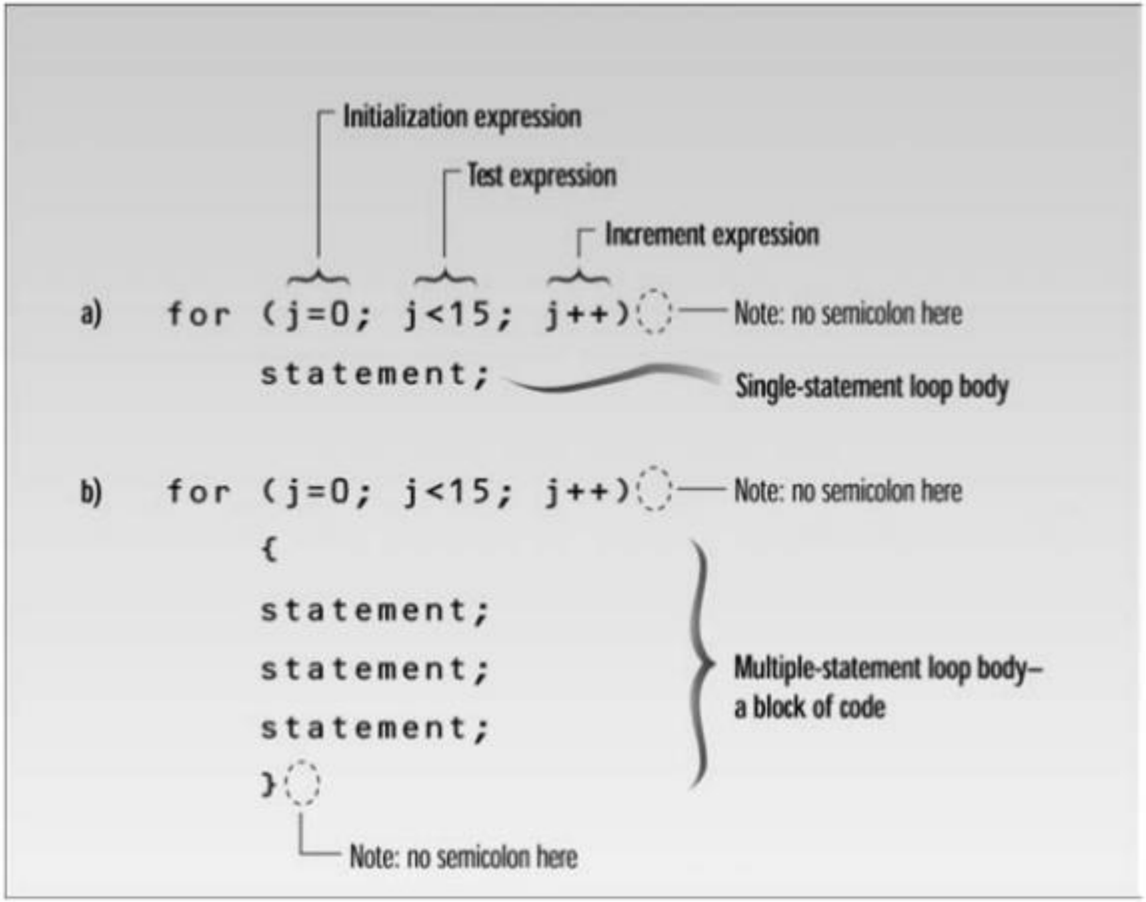
The for loop is the easiest C++ loop to understand. All its loop control elements are gathered in one place, while in the other loop constructions they are scattered about the program, which can make it harder to unravel how these loops work.

- The for loop executes a section of code a fixed number of times.



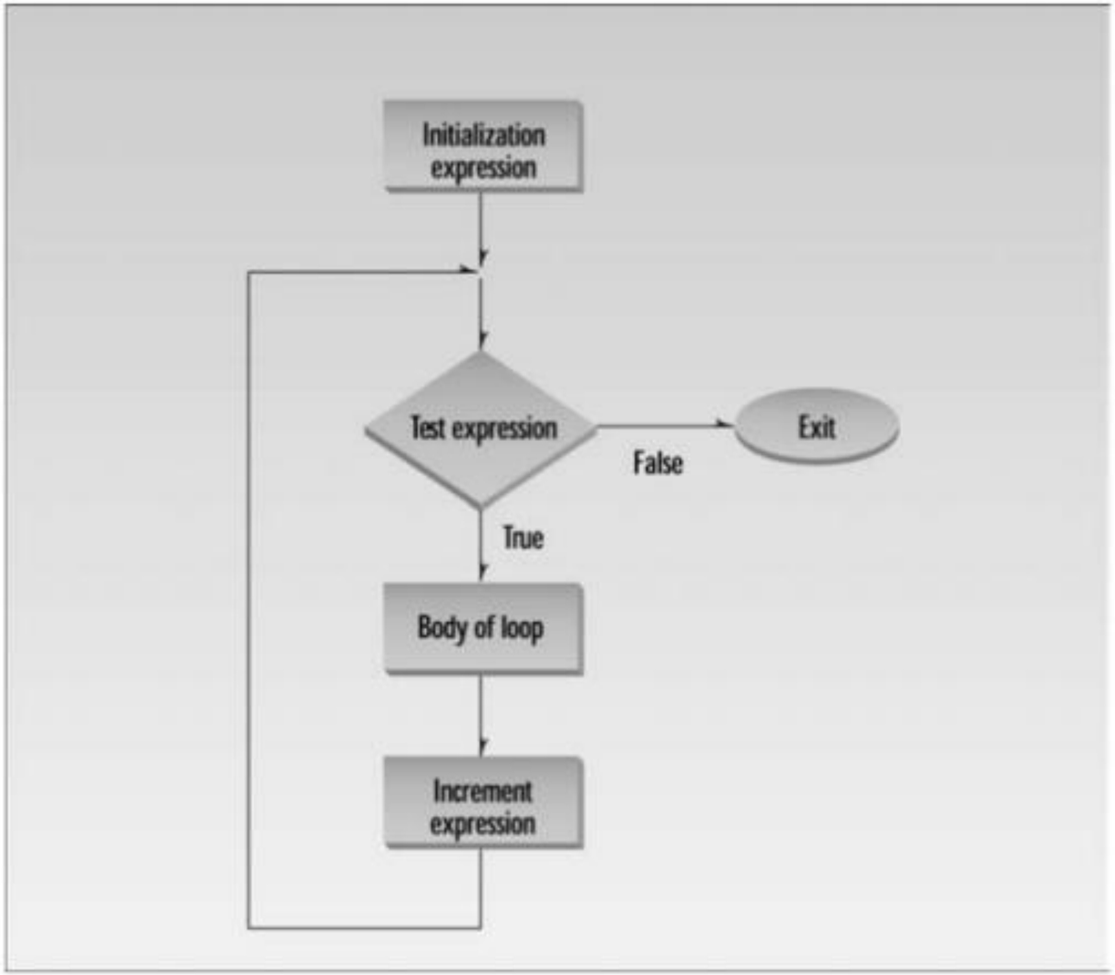
```
main(){
    int j;

    for(j = 0; j<15; j++)
        cout<< j * j <<endl;

    }
```

For loop Defination



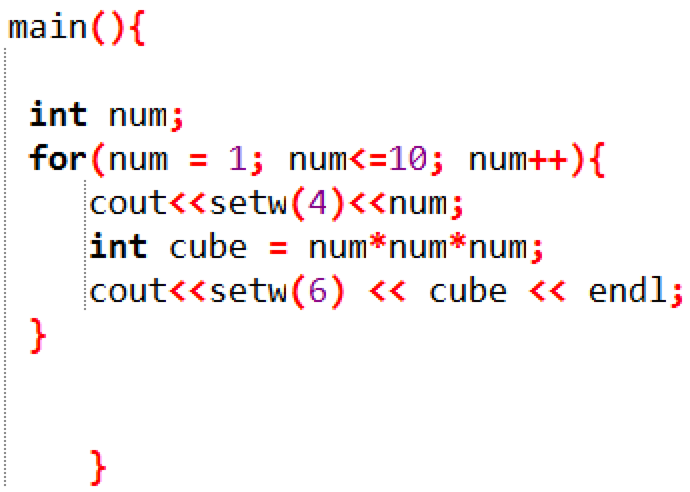
```
a)    for (j=0; j<15; j++)
          statement;

b)    for (j=0; j<15; j++)
          {
          statement;
          statement;
          statement;
          }
```

For loop Execution

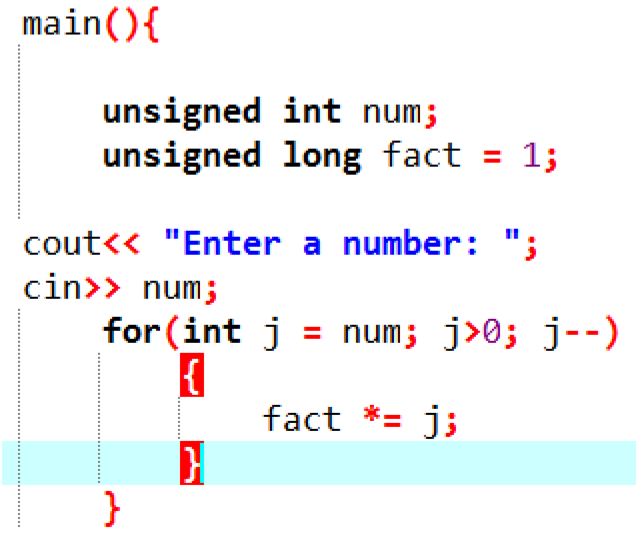


Multiple statements in the loop body

- Of course you may want to execute more than one statement in the loop body.
- Multiple statements are delimited by braces, just as functions are:

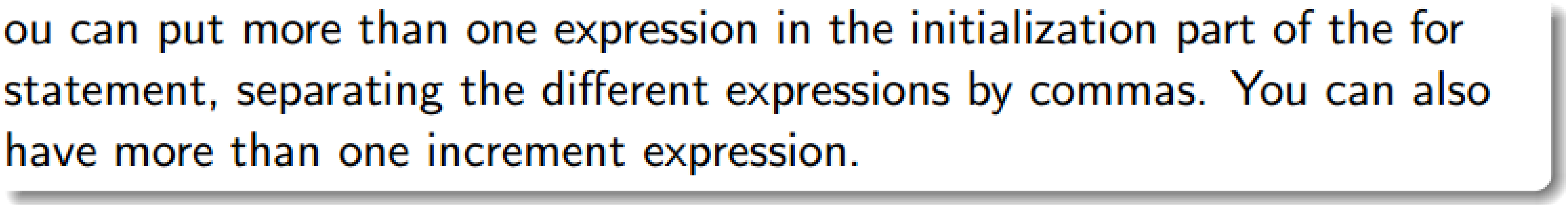


```
main(){

 int num;
 for(num = 1; num<=10; num++){
    cout<<setw(4)<<num;
    int cube = num*num*num;
    cout<<setw(6) << cube << endl;
 }


    }
```



```
main(){

    unsigned int num;
    unsigned long fact = 1;

cout<< "Enter a number: ";
cin>> num;
    for(int j = num; j>0; j--)
        {
            fact *= j;
        }
    }
```

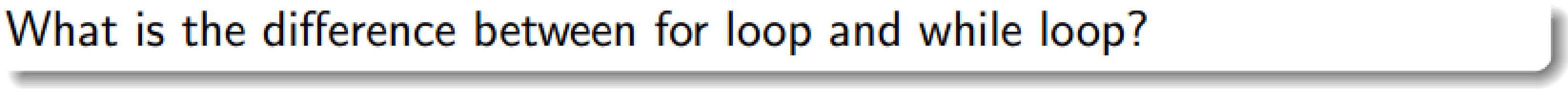
Multiple initialization and test expression



ou can put more than one expression in the initialization part of the for statement, separating the different expressions by commas. You can also have more than one increment expression.

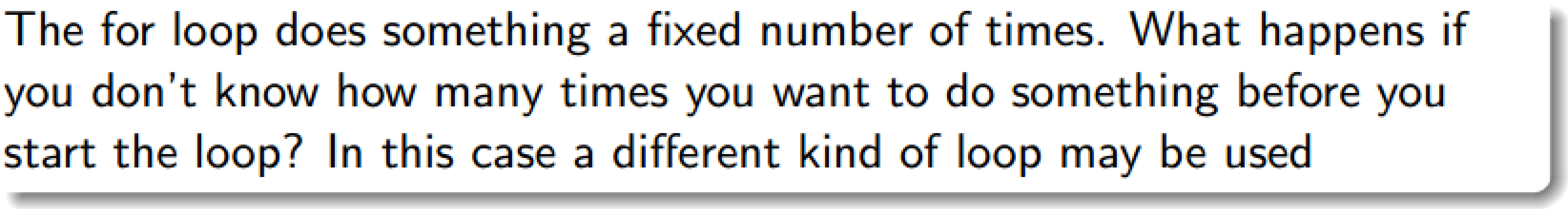
- For(j = 0, alpha = 100; j<50; j++, alpha--)
- You can have only one test expression.
- You can leave out some or all of the expressions if you want to. The expression for(;;).

Question



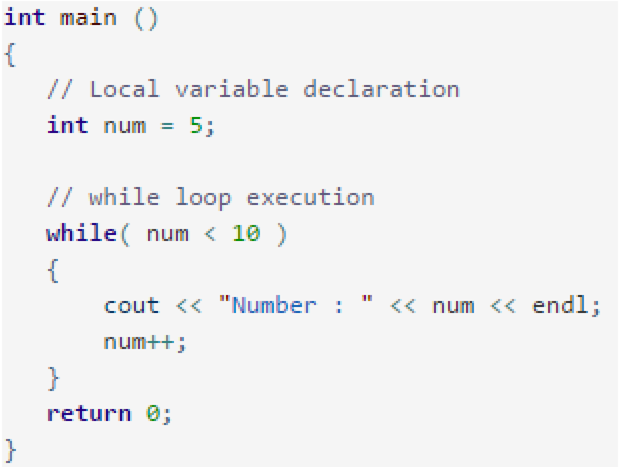
What is the difference between for loop and while loop?

The while loop



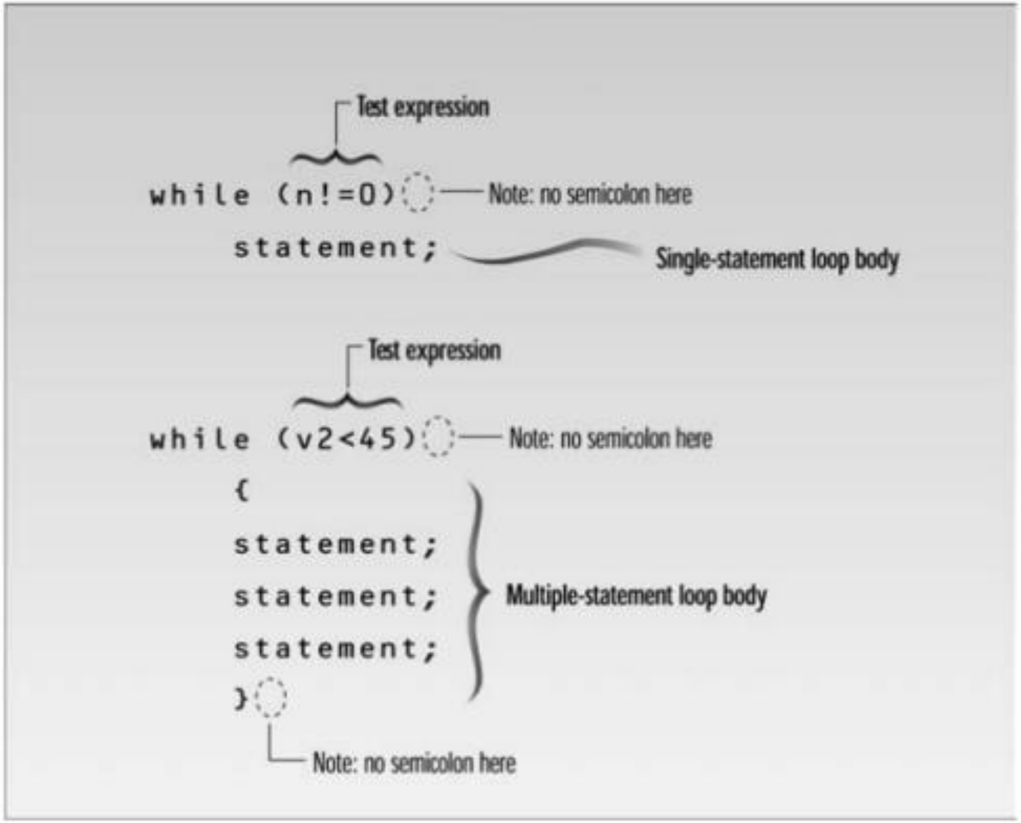
The for loop does something a fixed number of times. What happens if you don't know how many times you want to do something before you start the loop? In this case a different kind of loop may be used

- Variable must be declared and initialize before while loop.
- Variable is increment/decrement inside the body of while loop



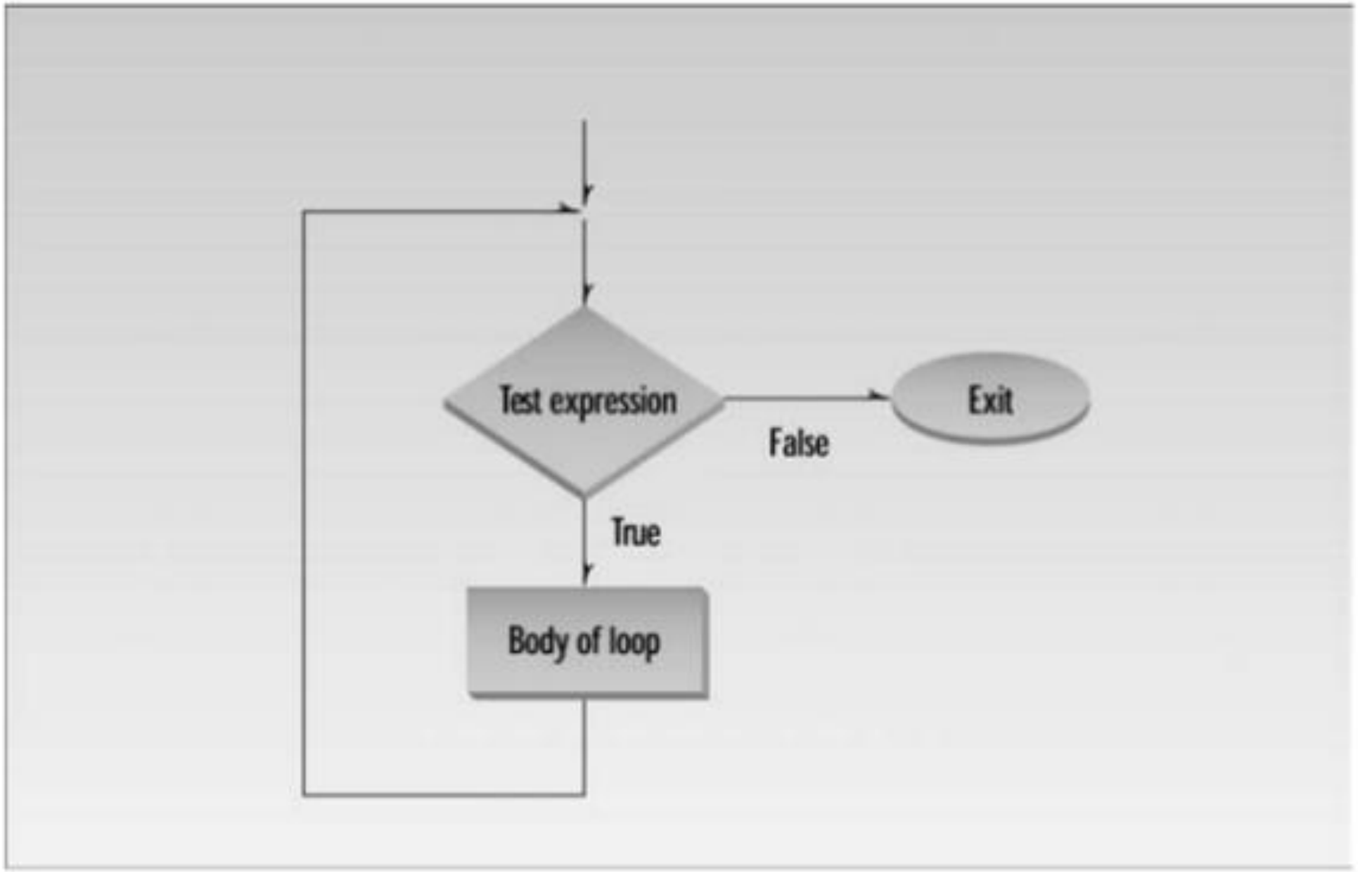
```
int main ()
{
   // Local variable declaration
   int num = 5;

   // while loop execution
   while( num < 10 )
   {
       cout << "Number : " << num << endl;
       num++;
   }
   return 0;
}
```

While loop operation

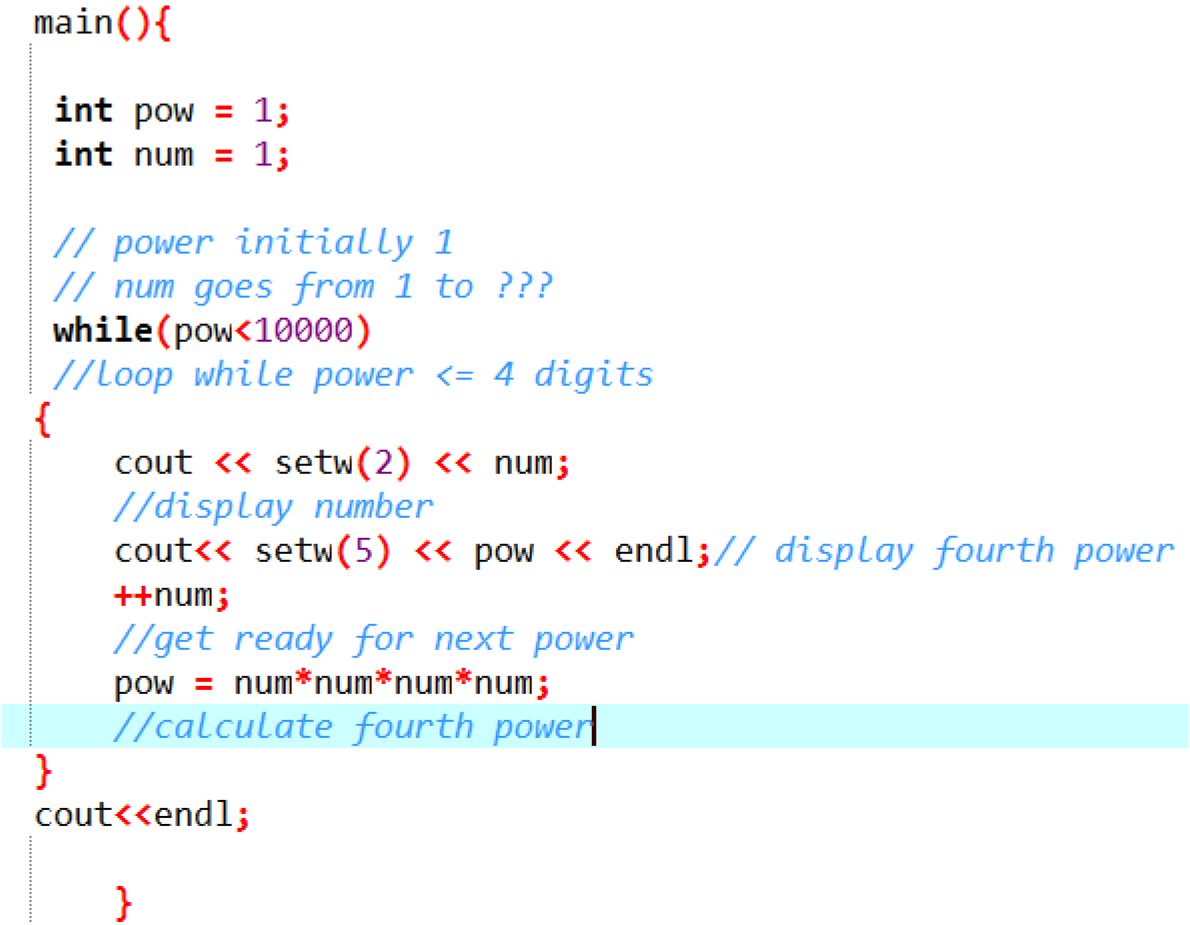


```
while (n!=0)
    statement;

while (v2<45)
    {
    statement;
    statement;
    statement;
    }
```

Condition Testing

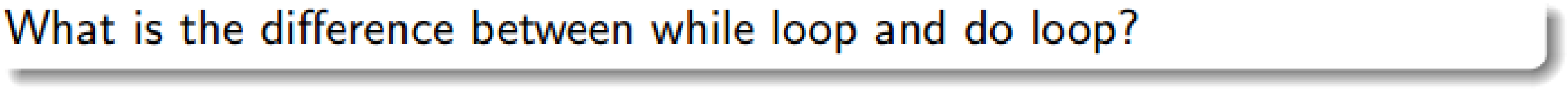


Example of while loop



```
main(){

 int pow = 1;
 int num = 1;

 // power initially 1
 // num goes from 1 to ???
 while(pow<10000)
 //loop while power <= 4 digits
{
    cout << setw(2) << num;
    //display number
    cout<< setw(5) << pow << endl;// display fourth power
    ++num;
    //get ready for next power
    pow = num*num*num*num;
    //calculate fourth power
}
cout<<endl;

    }
```

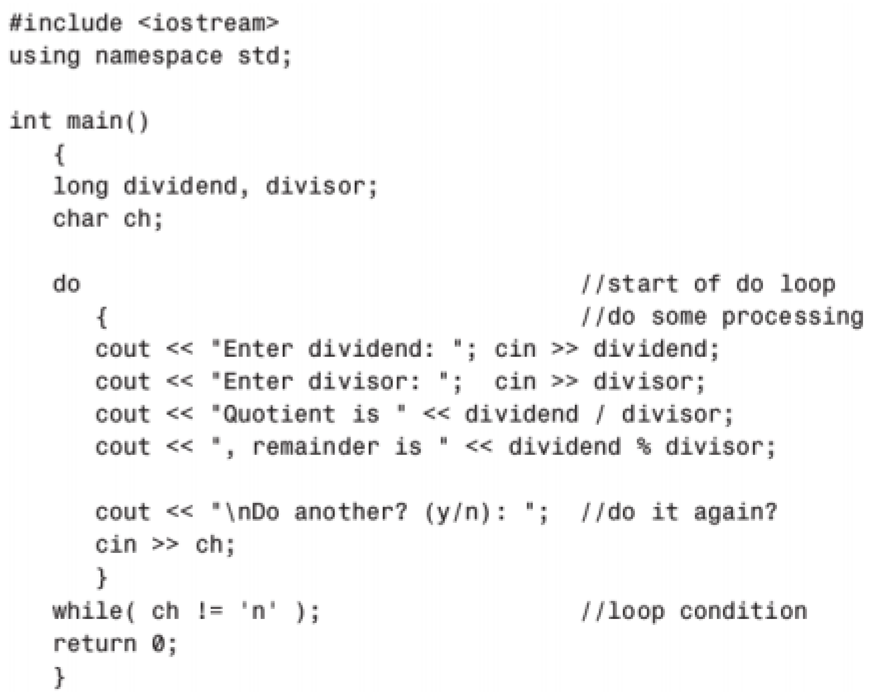
Question



What is the difference between while loop and do loop?

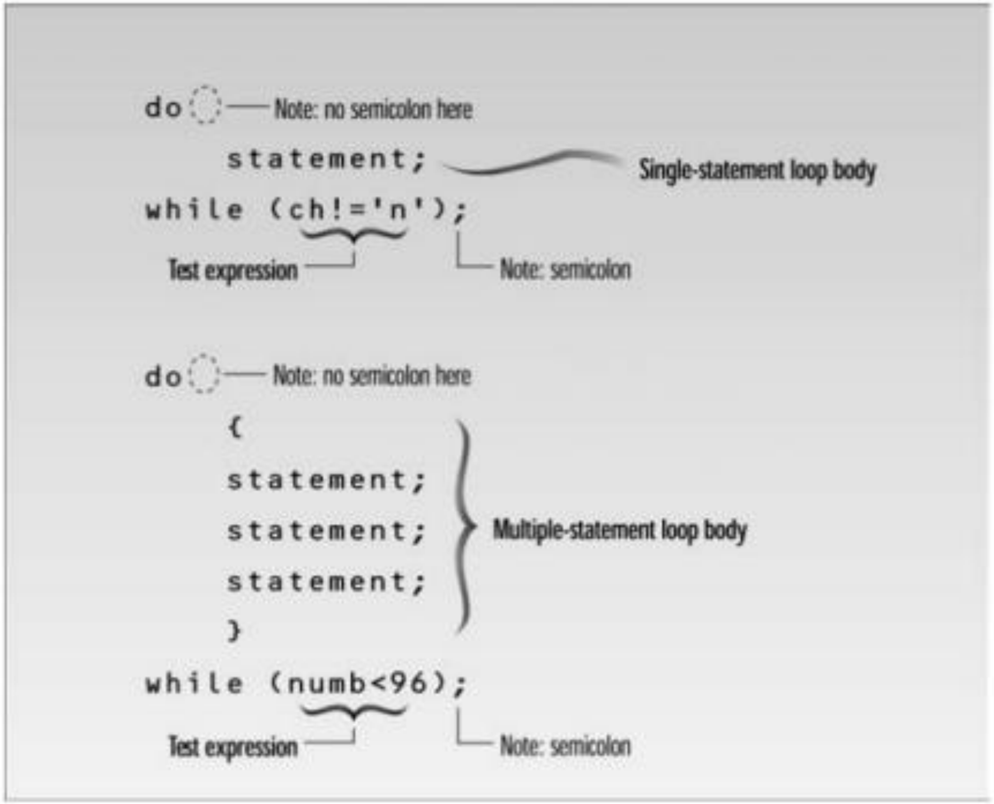
The do loop

Sometimes you want to guarantee that the loop body is executed at least once, no matter what the initial state of the test expression. When this is the case you should use the do loop, which places the test expression at the end of the loop.



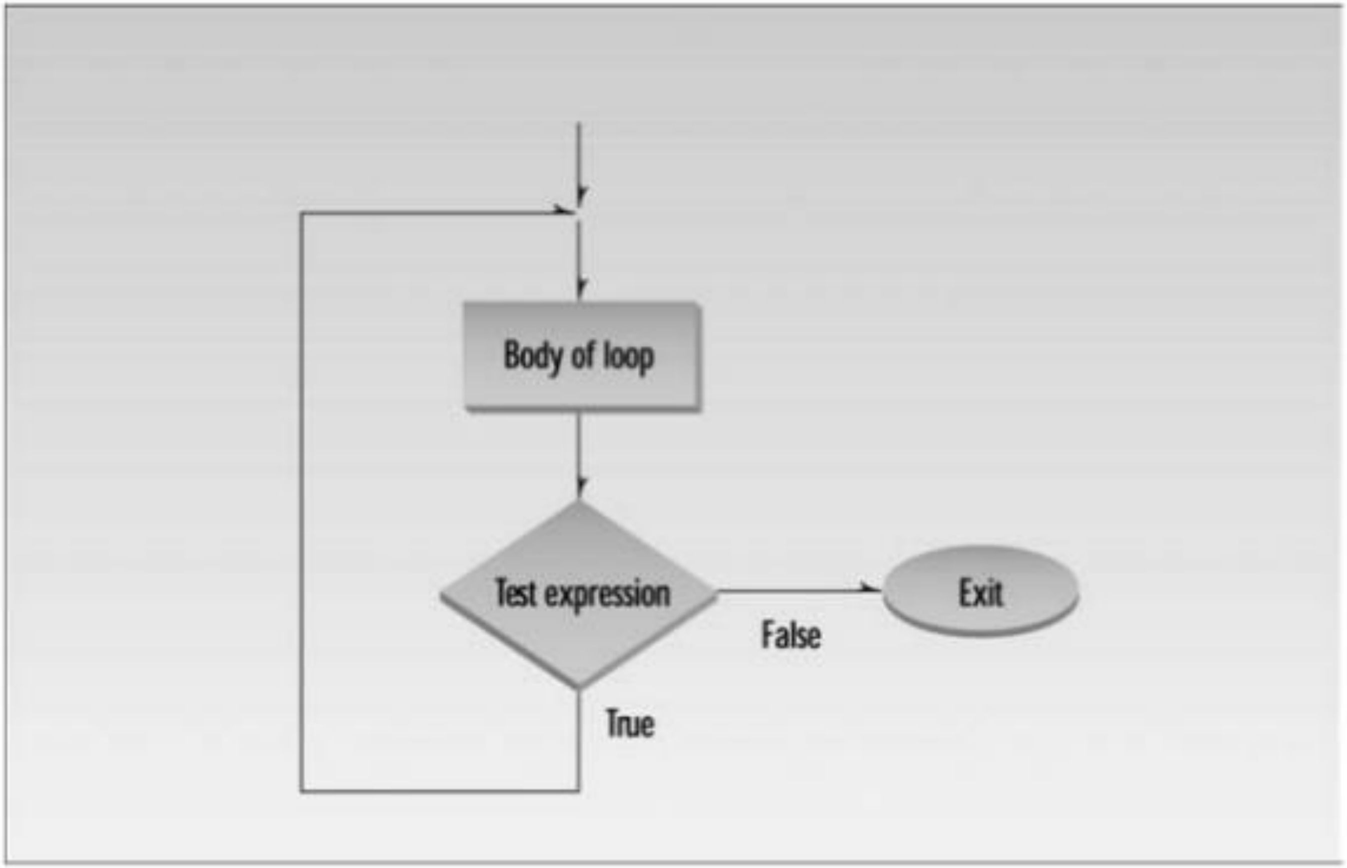
```
#include <iostream>
using namespace std;

int main()
   {
   long dividend, divisor;
   char ch;

   do                                    //start of do loop
      {                                  //do some processing
      cout << "Enter dividend: "; cin >> dividend;
      cout << "Enter divisor: ";  cin >> divisor;
      cout << "Quotient is " << dividend / divisor;
      cout << ", remainder is " << dividend % divisor;

      cout << "\nDo another? (y/n): ";   //do it again?
      cin >> ch;
      }
   while( ch != 'n' );                   //loop condition
   return 0;
   }
```

Writing do while loop



```
do
    statement;
while (ch!='n');

do
    {
    statement;
    statement;
    statement;
    }
while (numb<96);
```

Testing Condition in do while loop



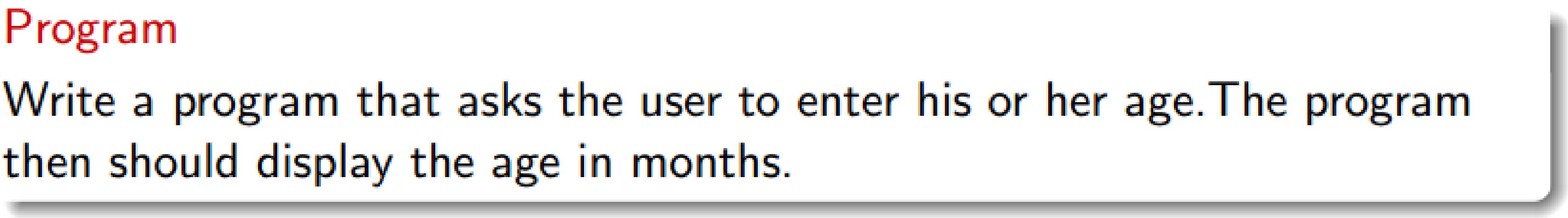
When to use which loop

- The for loop is appropriate when you know in advance how many times the loop will be executed

- The while and do loops are used when you do not know in advance when the loop will terminate

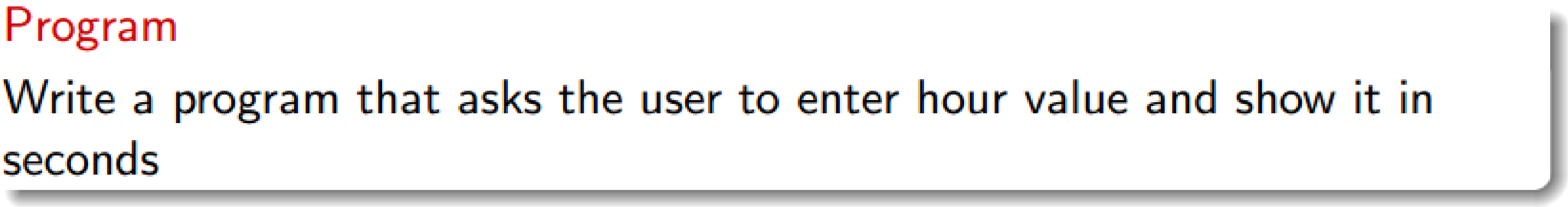
- Do while loop is used when at least once the execution of the program is important whether or not the condition is true.

Exercise-2



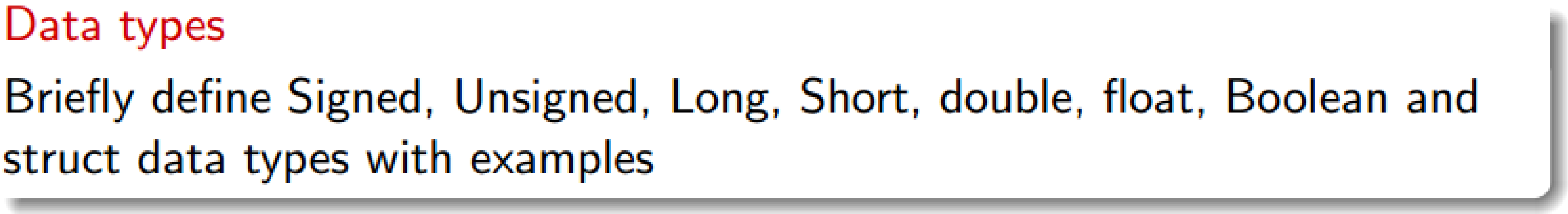
Program

Write a program that asks the user to enter his or her age. The program then should display the age in months.



Program

Write a program that asks the user to enter hour value and show it in seconds



Data types

Briefly define Signed, Unsigned, Long, Short, double, float, Boolean and struct data types with examples

Exercise-2

- **Program**
  - **Write a program that will print the following pattern:**

```
1******
12*****
123****
1234***
12345**
123456*
1234567
```

- **Program**
  - **Write a program that will ask the user to input n positive numbers. The program will terminate if one of those number is not positive.**

Any