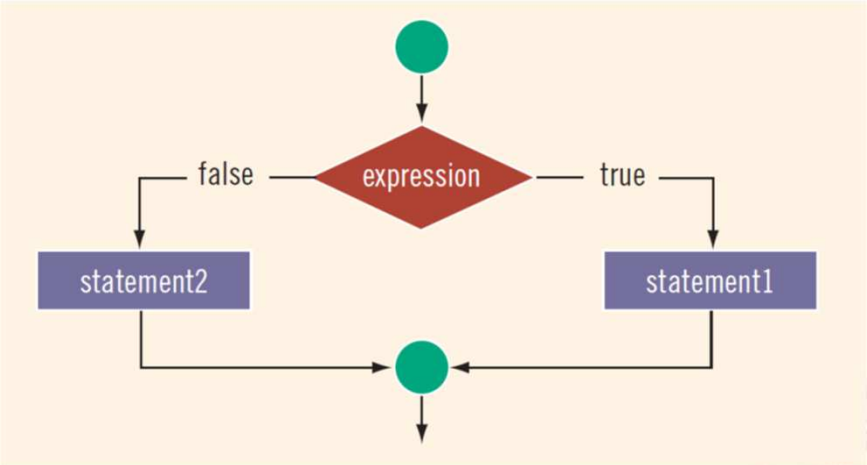
Lecture 5: C/C++ Control Structures

Dr. Mohammed Hawa
Electrical Engineering Department
University of Jordan

if … else …



Relational and Logical Operators

Operator	Description
==	Equal to
!=	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
&&	And
||	Or
!	Not

Example 1

```
double a = 3.12;
double b = 4.5;
double result;

// calculate the absolute value
if (a >= b)
{
	result = a - b;
}
else
{
	result = b - a;
}
```

- Do not use == and != for float and double since the number might not be stored exactly (using >, >=, <, <= is fine). For integers using == and != is safe.
- Avoid confusing the equality operator (==) and the assignment operator (=)

Example 2

```
int grade = 85;

if (grade == 100) // do not say: if (grade = 100)
       std::cout << "outstanding!" << std::endl;

else if(grade >= 90) // >= 90 but not 100 (why?)
       std::cout << "excellent!" << std::endl;

else if(grade >= 80) // >= 80 but < 90 (why?)
       std::cout << "very good!" << std::endl;

else if(grade >= 70) // >= 70 but < 80
       std::cout << "good!" << std::endl;

else if(grade >= 60) // >= 60 but < 70
       std::cout << "acceptable!" << std::endl;

else // < 60
       std::cout << "fail!" << std::endl;
```

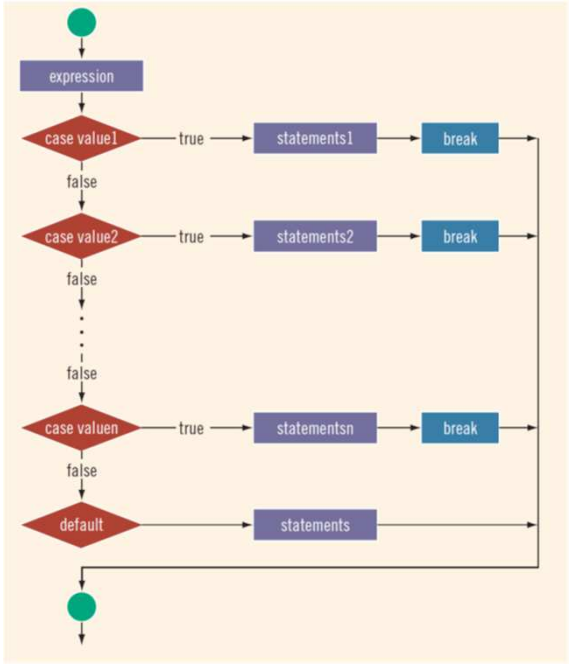

Homework

- Write C/C++ statements that output `Male` if the gender is 'M', `Female` if the gender is 'F', and `invalid gender` otherwise.
- What is the output of the following program?

```
int myNum = 10;
int yourNum = 30;
if (yourNum % myNum == 3) // modulo
{
       yourNum = 3;
       myNum = 1;
}
else if (yourNum % myNum == 2) // modulo
{
       yourNum = 2;
       myNum = 2;
}
else
{
       yourNum = 1;
       myNum = 3;
}

cout << myNum << " " << yourNum << endl;
```


switch

- Do not forget to use the break statement (*common mistake*)



Example (similar to if… else…)

```
char grade = 'B';

switch (grade)
{
	case 'A':
		std::cout << "The grade point is 4.0.";
		break;
	case 'B':
		std::cout << "The grade point is 3.0.";
		break;
	case 'C':
		std::cout << "The grade point is 2.0.";
		break;
	case 'D':
		std::cout << "The grade point is 1.0.";
		break;
	case 'F':
		std::cout << "The grade point is 0.0.";
		break;
	default:
		std::cout << "The grade is invalid.";
}
```

Repeat Something: `for`, `while`

```c
// count from 0 to 99
for (int i = 0; i < 100; i++)
{
        printf("%d\n", i);
}
```

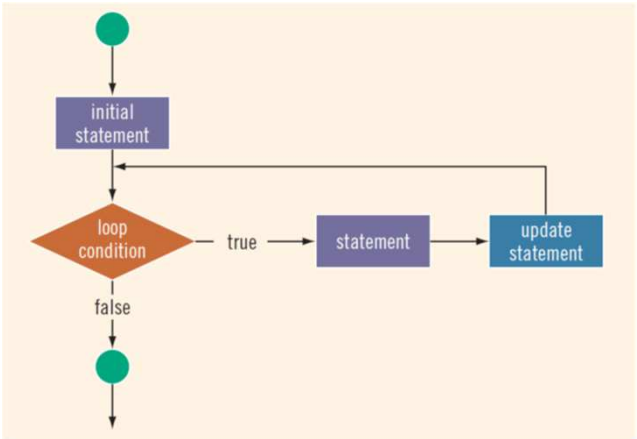
```c
// count backwards from 99 to 0
for (int j = 99; j >= 0; j--)
{
        printf("%d\n", j);
}
```

```c
// count from 0 to 99
int i = 0;
while (i < 100)
{
        printf("%d\n", i);
        i++;
}
```

```c
// count backwards from 99 to 0
int j = 99;
while (j >= 0)
{
        printf("%d\n", j);
        j--;
}
```

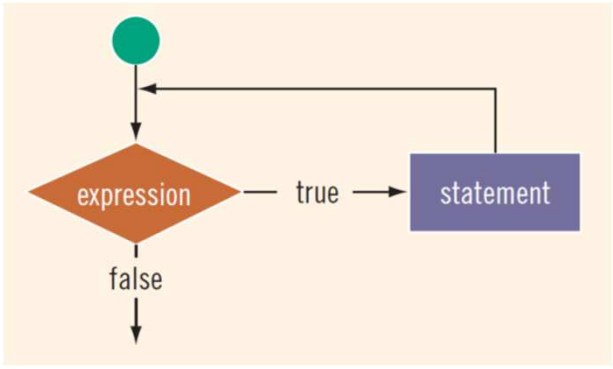

Details: `for`

```c
for (initial statement; loop condition; update statement)
    statement
```



Details: while

```
while (expression)
    statement
```

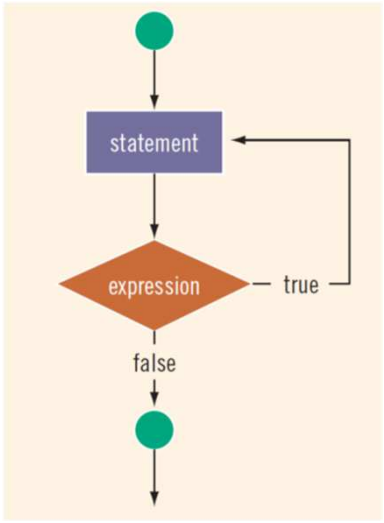


do … while

```
char password[20] = "secret";
char value[20];

do
{
        cout << "Enter password to unlock: ";
        cin >> value;

} while ( strcmp(value, password) != 0 );

cout << "Computer Unlocked :-)" << endl;
```


Details: do … while

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



Nested Loops

```
int i;
int j;

for (i = 1; i <= 5; i++) // outer loop
{
       for (j = 1; j <= i; j++) // inner loop
              std::cout << "*";

       std::cout << std::endl;
}
```

```
*
**
***
****
*****
```


`break` and `continue`

- Executing a `break` statement in the body of a loop immediately terminates the current loop (not all loops).
- Executing a `continue` statement in the body of a loop skips the current loop's remaining statements and proceeds with the next iteration.

Example

```
int x = 0;

while (x < 10)
{
	++x;

	if (x % 2 == 0) // skip even numbers
		continue;

	printf ("%i is an odd number.\n", x);
}
```

```
1 is an odd number.
3 is an odd number.
5 is an odd number.
7 is an odd number.
9 is an odd number.
```

Example

```
int    i;
float  grade;
float  sum = 0.0;
int    number = 0;

for(i = 1; true; i++)
{
       printf("\nEnter grades to find average (-1 to stop): ");

       scanf("%f",&grade);

       if(grade < 0.0)
              break; // exit the loop

       sum    += grade; // sum = sum + grade
       number += 1;  // number = number + 1
}

printf("\nAverage = %.2f", sum/number);
```

```
Enter grades to find average (-1 to stop): 78
Enter grades to find average (-1 to stop): 90.2
Enter grades to find average (-1 to stop): 85.5
Enter grades to find average (-1 to stop): 88
Enter grades to find average (-1 to stop): -1
Average = 85.43
```


C++11 foreach

```
vector<int> vec;
vec.push_back( 10 );
vec.push_back( 20 );
vec.push_back( 30 );

for (int i : vec ) // foreach
{
    cout << i;
}
```

