Introduction to Programming

Lecture 4: Statements and Control Flow

Mahmoud El-Gayyar

elgayyar@ci.suez.edu.eg

Review: Chapter 3

- *Types*
- *Constants*
- *Declarations*
  - Variable Names
- *Operators*
  - Arithmetic Operators
  - Assignment Operators
- *Function Calls*

Outline

- *Expression Statement*
- *Conditional*
  - if Statement
  - Nested if
  - switch Statement
- *Boolean Expressions*
- *Loops*
  - while Loop
  - for Loop
  - Continue & Break

Outline

- ***Expression Statement***
- *Conditional*
  - if Statement
  - Nested if
  - switch Statement
- *Boolean Expressions*
- *Loops*
  - while Loop
  - for Loop
  - Continue & Break

Expression Statements

- *Expression Statements:*
  - i = 0;
  - printf("Hello, world!\n");
- *Meaningless Expression Statements:*
  - 0;
  - i;
  - i+1;

Outline

- *Expression Statement*
- ***Conditional***
  - if Statement
  - Nested if
  - switch Statement
- *Boolean Expressions*
- *Loops*
  - while Loop
  - for Loop
  - Continue & Break

Conditional: IF Statement

- *The simplest way to modify the control flow (Branching)*

```
if(x > max)
        max = x;
```

```
if( expression )
        statement;
```

```
if( expression ){
        statement;
        statement;
        statement;
}
```

Conditional: IF Statement

- *What about the other branch: **else** clause*

```
if(n > 0)
      average = sum / n;
else{
      printf("can't compute average\n");
      average = 0;
}
```

```
if( expression )
            statement;        or    block
      else
            statement;        or    block
```

Conditional: IF Statement

- *Example: write an code fragment to check the value of a variable x and update the value of sign variable as follows:*
  - x<0 (sign=-1)
  - x=0 (sign= 0)
  - x>0 (sign= 1)

```
if (x < 0)
      sign = -1;
else{
      if (x == 0)
            sign = 0;
      else
            sign = 1;
```

```
if (x < 0)
   sign = -1;
if (x == 0)
   sign = 0;
if (x > 0)
   sign = 1;
```

Conditional: IF Statement

- *if-else statements can also be used to choose between several alternatives.*
  - *Suppose we have a student's numeric grade, and we want to print out the corresponding letter grade:*

100-90	A
89-80	B
87-70	C
69-60	D
< 60	F

```
if(grade >= 90)
        printf("A");
else if(grade >= 80)
        printf("B");
else if(grade >= 70)
        printf("C");
else if(grade >= 60)
        printf("D");
else
        printf("F");
```

Conditional: SWITCH Statement

- *switch can be used instead of if-else to choose between several alternatives:*
  - Check for exact values of a certain variable. For example the user will input an integer from 1 .. 7 and you will print the corresponding week day (1→ Friday, 2 → Saturday, ..)

```
switch ( day ) {
   case 1 : printf("Sunday");
            break;
   case 2 : printf("Monday");
            break;
        ...
   case 6 : printf("Friday");
            break;
   case 7 : printf("Saturday");
            break;
  default : printf("Not an allowable day number");
  }
```

Conditional: SWITCH Statement

```
switch (selector){
      case label1: statement1;
          break;
      case label2: statement2;
          break;
          ...
      case labeln: statementn;
          break;
      default: statementd;                    // optional
}
```

Conditional: SWITCH Statement

```
switch (day){
    case 1 :
    case 7 : printf("This is a weekend day");
             break;
    case 2 :
    case 3 :
    case 4 :
    case 5 :
    case 6 : printf("This is a weekday");
             break;
    default : printf("Not a legal day");
  }
```

Outline

- *Expression Statement*
- *Conditional*
  - if Statement
  - Nested if
  - switch Statement
- ***Boolean Expressions***
- *Loops*
  - while Loop
  - for Loop
  - Continue & Break

Boolean Expression

- *Boolean Expression → True / False*
- *In C integers can be used as boolean expressions*
  - Non-zero values (negative or positive) → True
  - Zero → False

RelationalOperator	Function
<	Lessthan
<=	Lessthanorequal
>	Greaterthan
>=	Greaterthanorequal
==	Equal
!=	Notequal

Boolean Expression

- *The most common error in C*


```
if(a = 0)
```

```
if(0 == a)
```

- *What about if you would like to make a complex boolean expression:*
  - **Example 1:** to test if x between 10 and 20.
  - **Example 2:** to test if x less than 10, or greater than 20

Boolean Expression

Boolean Operator	Function
&&	Logical AND
||	Logical OR
!	Logical NOT(Unary operator; Takes one operand)

- ***Example 1:*** *to test if x between 10 and 20.*
  - **x >= 10 && x <= 20**

10 <= x <=20

- ***Example 2:*** *to test if x less than 10, or greater than 20*
  - ?!!

Outline

- *Expression Statement*
- *Conditional*
  - if Statement
  - Nested if
  - switch Statement
- *Boolean Expressions*
- ***Loops***
  - while Loop
  - for Loop
  - Continue & Break

Repeatition: Loops

- *Loops allow you to repeat a set of instructions. In most of the cases, it need a **control variable** to control the loop and four main components:*
  - ***Initial value*** (of the control variable)
  - ***Control expression*** (the loop repeats as long as that expression is true)
  - ***Body*** (the statements to repeat)
  - ***Step*** (required to reach the state where the control expression will be false and the loops will terminate, otherwise, we will go into ***infinite*** loop)

Repeatition: While Loop

- *The most basic loop in C is the while loop.*

```
while( expression )
            statement;        or    block
```

```
int x = 2;
while(x < 1000){
      printf("%d\n", x);
      x = x * 2;
}
```

Initial Value

Control Expression

Step

//very important

Repeatition: For Loop

- *The second loop that we have mentioned already:*

```
for( expr1; expr2 ; expr3)
      statement   or    block;
```

```
int x = 2;
while(x < 1000){
      printf("%d\n", x);
      x = x * 2;              //very important
}
```

```
for(int x = 2; x < 1000; x = x * 2)
      printf("%d\n", x);
```

Repeatition: For Loop

- *The three expressions are optional*

```
exp1;
for( ; expr2 ; expr3)
      statement  or    block;
```

```
expr1;
for( ; expr2 ; ){
      statement;
      exp3;
}
```

```
for( ; ; ){
      statement  or    block;
}
```

```
while (1){
      statement;
}
```

Continue and Break

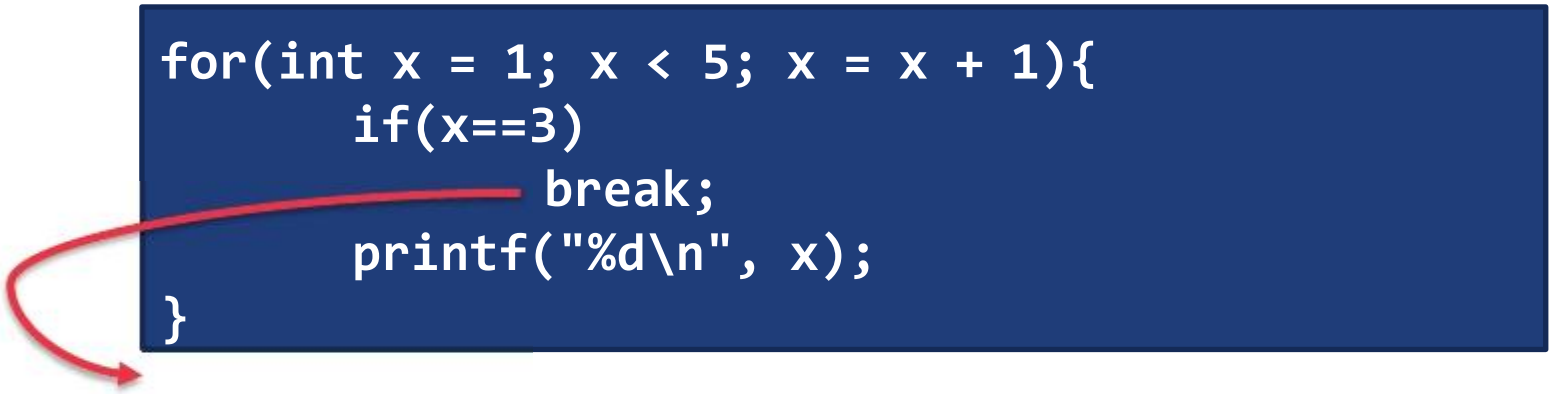
- *continue **skips** the current iteration of a loop and proceeds directly to the next iteration*

```
for(int x = 1; x < 5; x = x + 1){
        if(x==3)
                continue;
        printf("%d\n", x);
}
```

```
int x = 1;
While(x < 5){
        if(x==3)
                continue;
        printf("%d\n", x);
        x= x + 1;
}
```

Continue and Break

- *break **terminates** (exists / stops) a loop*



Full Example: Print Prime Numbers between 1 & 100

```
int main(){
	int i, j;
	printf("%d\n", 2);

	for(i = 3; i <= 100; i = i + 1){
		for(j = 2; j < i; j = j + 1){

			if(i % j == 0)
				break;

			if(j > sqrt(i)){
				printf("%d\n", i);
				break;
			}
		}
	}
	return 0;
}
```

Summary

- *How Branch?*
  - if-else statement
  - switch statement
- *Complex logical expressions*
- *How to loop?*
  - while
  - for
- *How to exist or skip an iteration in a loop?*
  - Break
  - continue