
University of Engineering and Technology,
Peshawar, Pakistan

# CSE 102: Computer Programming

## Lecture 3

# Decision Making and Loops

By;
Dr. Muhammad Athar Javed Sethi

# The If Statement

- Syntax:  if (expression) statement;
  - If the expression is true (not zero), the statement is executed.
  - If the expression is false, it is not executed.
- You can group multiple expressions together with braces:

```
if (expression) {
  statement 1;
  statement 2;
  statement 3;
}
```

# Logic Operators in ‘if’

- Equal to `if (x==10)`
- Not equal to `if (x!=10)`
- Less than `if (x<10)`
- Greater than `if (x>10)`
- Less than / equal to `if (x<=10)`
- Greater than / equal to `if (x>=10)`

# Compound Operators

- Logical AND
- Logical OR
- Logical NOT

```
if (x==1 && y==2)
if (x==1 || y==2)
if (!x) …
```

# If else

```
if(income > 17000)
    printf(“pay tax”);
else
    printf(“find a better job”);
```

one of these statements always execute

# Single and Compound Statements

**Single statements:**

```
if (condition)
  true_statement;
else
  false_statement;
```

**Multiple statements:**

```
if (condition)
{
   ……
}
else
{
   ……
}
```

# Nested If Statements

```
int main(void)
using namespace std;
{
   int winner = 1;
   cout << "…and the winner of ICC is ";
   if (winner==1)
     cout << "Pakistan";
   else if (winner==2)
     cout << "England";
   else if (winner==3)
     cout << "WI";
   else
     cout << "Australia";

 }
```

# Switch Statements

- Switch statements look like this example:

```
 switch (expression)
{
      case value_1 : statements_1; break;
      case value_2 : statements_2; break;
      ...
      case value_n : statements_n; break
      default:
 }
```

# Loops

- Repeat a series of statements
- Not reasonable to copy statements multiple time
- Need a way to repeat a block of code
- Loops allow repetition

# The For Loop

- Syntax:
- ```
  for (initialization; test; increment)
      {
          statements;
      }
  ```
- The for loop will first perform the initialization. Then, as long test is TRUE, it will execute statements. After each execution, it will increment.
  ```

# For loop

- Known number of iterations

```
for(count=1;count<=10;count++)
{
    body of loop
}
```

# For loop examples

```
for (x=20 ; x <= 80; x +=10)
```

- from 20 to 80 in steps of 10

```
for(x=80; x >= 20; x -=10)
```

- from 80 to 20 in steps of -10

# While loop

```
degree =0;
while (degree <= 360)
{
    degree += increment;
}
```

# The While Loop

- An example while loop looks like this:

```
Using namespace std;

int main()
{
  char ch;

  while (ch != 'Q')
  {
     …
      cin>>ch;
  }
 return 0;
```

# The Do-while Loop

- The do-while loop repeatedly executes a block of code indicated by statements as long as the conditional expression *cond_expr* is true.

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

# A ‘while’ for ‘for’

```
i=0;
while(i<10)
{
   body of the loop;
   i++;
}
```

is equivalent to

```
for(i=0; i<10; i++)
{
  body of the loop;
}
```

# A ‘for’ for ‘while’

```
for(; degree<360;)
{
   degree += increment;
}
```

# Special Cases

- You can have as many control variables as you want in loops. The following is fine:

```
for (x=0, y=0; x+y<10; x++, y++)
```

# A forever loop

- Using for

```
for(;;)
{
  Cout << “Hello\n”;
}
```

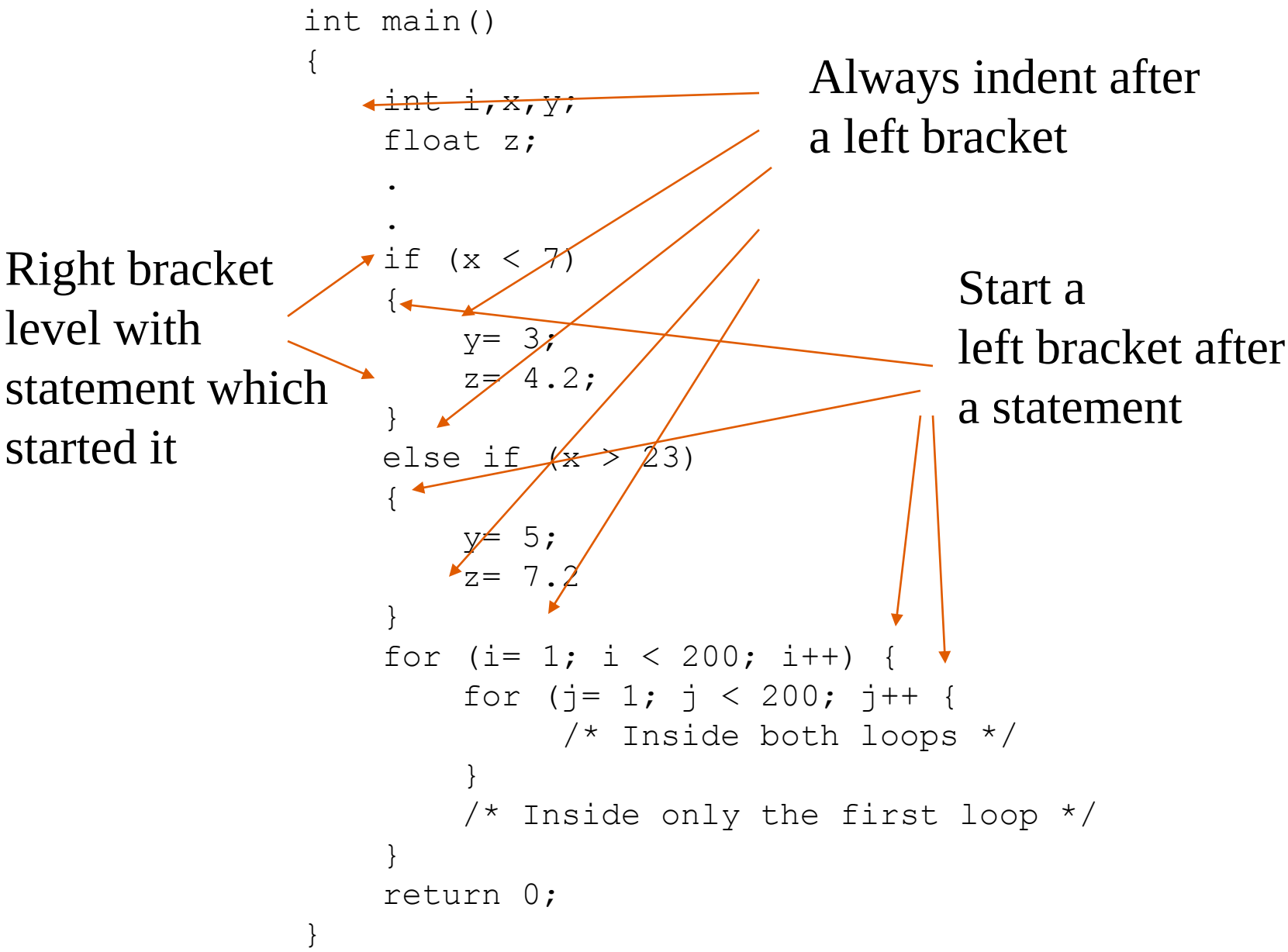
- Using while

```
while(1)
{
  Cout<< “This is C++ Program\n”);
}
```

# Continue & Break

- The **continue** statement shifts the control back to the beginning of the loop. It is used inside the body of loop.
- The **break** statement is used to halt execution of a loop prior to the loop's normal test condition being met.
- The **exit** statement causes the whole program to terminate if it is called within the main program block.

# Coding for readability




```
int main()
{
    int i,x,y;
    float z;
    .
    .
    if (x < 7)
    {
        y= 3;
        z= 4.2;
    }
    else if (x > 23)
    {
        y= 5;
        z= 7.2
    }
    for (i= 1; i < 200; i++) {
        for (j= 1; j < 200; j++ {
            /* Inside both loops */
        }
        /* Inside only the first loop */
    }
    return 0;
}
```