Repetition Statement

Computer Engineering

Yusramohammed@tiu.edu.iq

2022 - 2023

Compound Assignment Operators

- We can do mathematical operations, +, -, *, / or % , in the form like

variable operator= expression; Example `sum+=3;`

variable = variable operator expression; Example `sum=sum+3;`

Assignment operator	Sample expression	Explanation	Assigns
Assume: `int c = 3, d = 5, e = 4, f = 6, g = 12;`			
+=	`c += 7`	`c = c + 7`	10 to c
-=	`d -= 4`	`d = d - 4`	1 to d
*=	`e *= 5`	`e = e * 5`	20 to e
/=	`f /= 3`	`f = f / 3`	2 to f
%=	`g %= 9`	`g = g % 9`	3 to g

```
int sum=1;
System.out.println("Sum is: "+sum);
sum+=3;
System.out.println("Sum is: "+sum);
```

```
int sum=1;
System.out.println("Sum is: "+sum);
sum=sum+3;
System.out.println("Sum is: "+sum);
```

output

```
run:
Sum is: 1
Sum is: 4
```

- **Example**

```
int c=10;
System.out.println("C= "+c);
c=c+1;
System.out.println("After adding 1\nC= "+c);
c=c-4;
System.out.println("After subtracting 4 \nC= "+c);
c=c*5;
System.out.println("After multiplying 5\nC= "+c);
c=c/2;
System.out.println("After dividing by 2\nC= "+c);
```

output

```
run:
C= 10
After adding 1
C= 11
After subtracting 4
C= 7
After multiplying 5
C= 35
After dividing by 2
C= 17
```

Compound Assignment Operators

- We can also do the operations by any value entered by user, for example.
- **Example**

```
 Scanner input=new Scanner(System.in);
 int c=10;
 System.out.println("C= "+c);
System.out.println("Enter a number: ");
int number=input.nextInt();
c+=number;
System.out.println("After adding the number to C \nC= "+c);
c-=number;
System.out.println("After subtracting number from C \nC= "+c);
c*=number;
System.out.println("After multiplying number to C\nC= "+c);
c/=number;
System.out.println("After dividing by number\nC= "+c);
```

```
run:
C= 10
Enter a number:
4
After adding the number to C
C= 14
After subtracting number from C
C= 10
After multiplying number to C
C= 40
After dividing by number
C= 10
```

Example

- Write a java program to ask the student to enter his/her grade. If the mark is larger and equal to 70 give 5 extra mark to his grade, otherwise, don't change the grade.

```
Scanner input=new Scanner(System.in);
System.out.print("Enter your grade: ");
int grade=input.nextInt();
if(grade>=70){
    grade+=5; //add 5 to the grade and save the new value of grade
    System.out.println("Your grade is: "+grade);
}
else{
    System.out.println("No extra Mark. Study harder");
    System.out.println("Your grade is: "+grade);
}
```

```
run:
Enter your grade: 60
No extra Mark. Study harder
Your grade is: 60
```

OR

```
run:
Enter your grade: 79
Your grade is: 84
```

Increment and Decrement Operators

- Java provides two unary operators (summarized in the table) for adding 1 to or subtracting 1 from the value of a numeric variable.
- These are the unary increment operator, **++,** and the unary decrement operator, **--**
- A program can increment by 1 the value of a variable called c using the increment operator, ++, rather than the expression c = c + 1 or c += 1.
- An increment decrement operator that's prefixed to (placed before) a variable is referred to as the ***prefix increment*** and known as ***preincrementing***

++	prefix increment	++a	Increment a by 1, then use the new value of a in the expression in which a resides.

- A decrement operator that's prefixed to (placed before) a variable is referred to as the ***prefix decrement operator*** and known as ***predecrementing***

--	prefix decrement	--b	Decrement b by 1, then use the new value of b in the expression in which b resides.

Increment and Decrement Operators

- An Increment operator that's postfixed to (placed after) a variable is referred to as the ***postfix increment*** and known as ***postincrementing***

++	postfix increment	a++	Use the current value of a in the expression in which a resides, then increment a by 1.

- A decrement operator that's postfixed to (placed after) a variable is referred to as the ***postfix decrement operator*** and known as ***postdecrementing*** .

--	postfix decrement	b--	Use the current value of b in the expression in which b resides, then decrement b by 1.

- We can increment or decrement a value by 1 in three different ways for example adding 1 to the variable a be like:
  - a=a+1 ; a+=1; a++;
  - a=a-1; a-=1; a--;

Increment or Decrement (Example)

- ***Prefix increment***

```
int a=0;
System.out.println(a);
System.out.println(++a);
System.out.println(a);
```

```
run:
0
1
1
```

- ***postfix increment***

```
int a=0;
System.out.println(a);
System.out.println(a++);
System.out.println(a);
```

```
run:
0
0
1
```

- ***prefix decrement***

```
int a=6;
System.out.println(a);
System.out.println(--a);
System.out.println(a);
```

```
run:
6
5
5
```

- ***postfix decrement***

```
int a=6;
System.out.println(a);
System.out.println(a--);
System.out.println(a);
```

```
run:
6
6
5
```

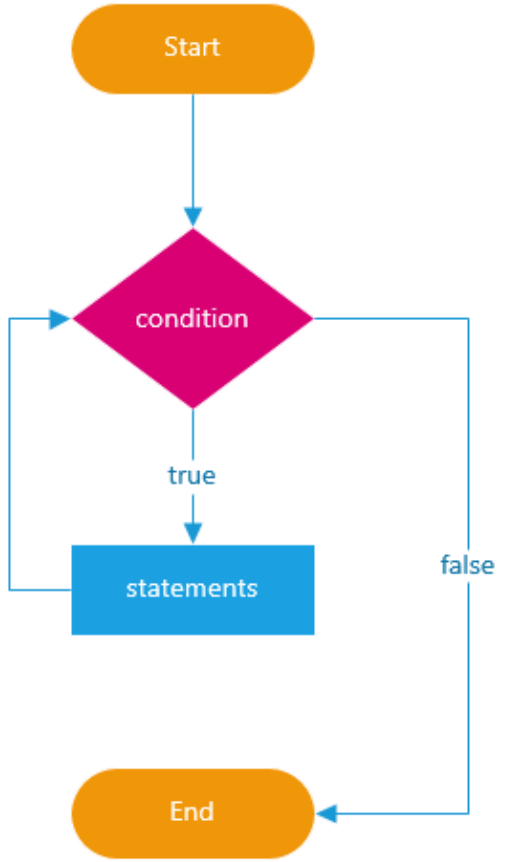
While Repetition statement

- A **repetition** (or **looping**) **statement** allows you to specify that a program should repeat an action while some condition remains true.
- The loop statement allow the program to execute a statement or a block of statements several times depending of the condition
- If the number of iterations are not fixed, it is recommended to use while loop.
- **The syntax structure of while statement is:**

```
Declaration and Initialization
While (condition)
{
      //Your Code
      Increment or decrement

}
```

```
int x=1;
While (x<10)
{
      System.out.println(x);
      x++;
}
```



- If the condition is true the statements is executed repeatedly until the condition is false

While Repetition statement

- **Example**: Write a program to print numbers between 1 to 10 inclusive using while loop.
- Declare a variable x and set its value to 1.
- While the value is smaller than or equal to 0, print the number.
- Increment the value of x by 1.

```
public class While {
    public static void main(String[] args) {
        System.out.println("The numbers between 1 to 10: ");
        int x=1;
        while(x<=10)
        {
            System.out.println(x);
            x++;
        }
    }
}
```

```
run:
0
1
2
3
4
5
6
7
8
9
10
```

While Repetition statement

- **Example**: Write a program to print numbers between 0 to 5 inclusive backwards using while loop.
- Declare a variable x and set its value to 5.
- While the value is larger than or equal to 10, print the number.
- Decrement the value of x by 1.

```
int x=5;
while(x>=0)
{
    System.out.println(x);
    x--;

}
```

```
run:
5
4
3
2
1
0
```

While Repetition statement

- **Example**: Write a java program to count the odd and even numbers between 1 to 100 using while loop.

```
int odd=0;
int even=0;
int jar=1;
while(jar<=100){
    switch(jar%2){
        case 1:
            odd++;
            break;
        case 0:
            even++;
            break;
    }
    jar++;
}System.out.println("no of odds: "+odd);
    System.out.println("no of evens: "+even);
```

```
run:
no of odds: 50
no of evens: 50
```

While Repetition statement

- **Example**: Write a java program to ask the user to enter 8 numbers and find the sum of the numbers and multiply the sum by 3, using while loop.

```
Scanner input=new Scanner(System.in);
int number;
int sum=0;
int count=1;
int result;
System.out.println("Enter 8 numbers: ");
while (count<=8) {
    number=input.nextInt();
    sum+=number;
    count++;
}
result=sum*3;
System.out.println("Sum is --> "+sum);
System.out.println("Result of "+sum+" * "+3+" is --> "+ result);
```

```
run:
Enter 8 numbers:
4
3
2
4
2
7
6
3
Sum is --> 31
Result of 31 * 3 is --> 93
```

Formulating Algorithms: Counter – Controlled Repetition

- In this repetition type, the number of inputs are known. This counter is assigned from the programmer and user.
- We use **counter-controlled repetition** to input data one at a time.
- Counter-controlled repetition is often called **definite repetition**, because the number of repetitions is known *before* the loop begins executing.
- **Example:** Write a java program to sum the series of numbers entered by the user. The user assigns the number counts.

```
Scanner input=new Scanner(System.in);
int sum=0;
System.out.println("howmany numbers do you have:");
int numbers=input.nextInt();
System.out.println("Enter numbers: ");
int jar=1;
while(jar<=numbers)
{
    int number=input.nextInt();
    sum+=number;
    jar++;
}
System.out.println("Sum of numbers is: "+sum);
```

```
run:
howmany numbers do you have:
4
Enter numbers:
5
3
2
6
Sum of numbers is: 16
```

Formulating Algorithms: Sentinel-Controlled Repetition

- **Sentinel-controlled repetition** is often called **indefinite repetition** because the number of repetitions is *not* known before the loop begins executing.
- The number of inputs are not known. It is assigned by user.
- The programmer set a fixed value to stop the program. Such as 0, stop, exit.
- **Example:** *Develop a summation program that processes sum for a series of arbitrary numbers each time it's run. The program should stop when the user enters -1*

```
Scanner input=new Scanner(System.in);
int sum=0;

System.out.println("Enter numbers or -1 to exit: ");
int number=input.nextInt();
while(number!=-1)
{
    sum+=number;
    number=input.nextInt();
}
System.out.println("Sum of numbers is: "+sum);
```

```
run:
Enter numbers or -1 to exit:
3
4
5
-1
Sum of numbers is: 12
```

Sample Development

- Write a java program to add the price of arbitrary number of products entered by the user. The user assign the number of products to be entered. Then discount the price by 10% if it is larger or equal to 100, otherwise don’t make any change to the price.