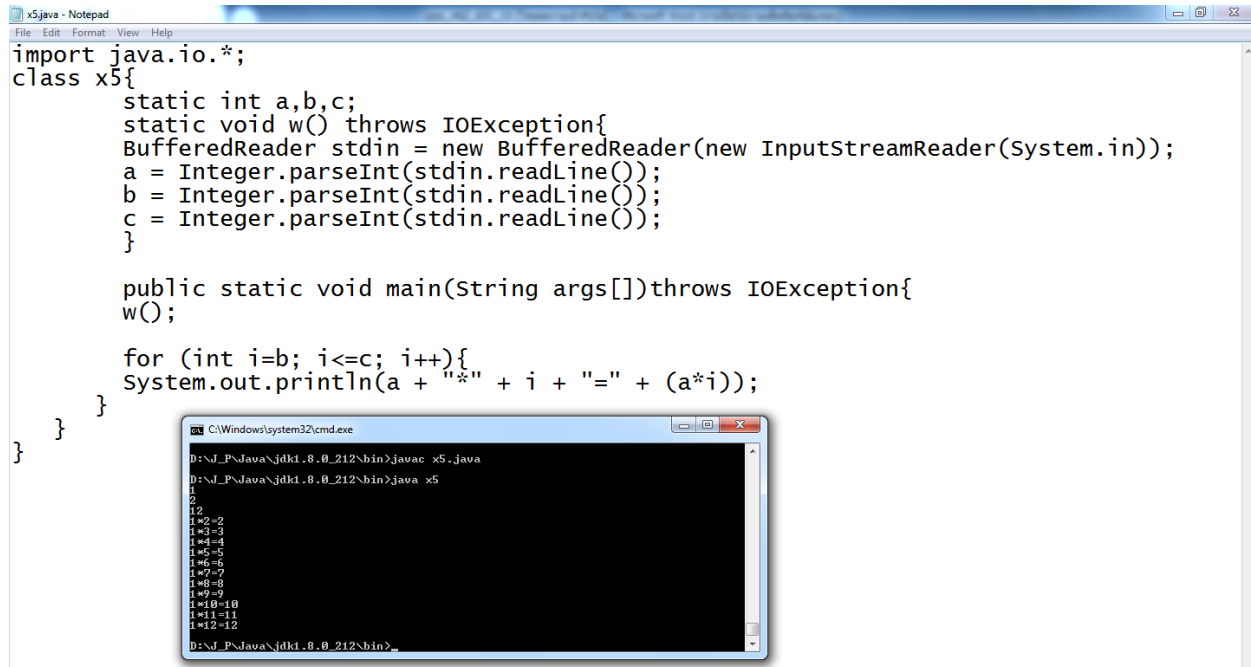


1.รับค่าจาก command line



```
import java.io.*;
class x5{
    static int a,b,c;
    static void w() throws IOException{
        BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));
        a = Integer.parseInt(stdin.readLine());
        b = Integer.parseInt(stdin.readLine());
        c = Integer.parseInt(stdin.readLine());
    }

    public static void main(String args[])throws IOException{
        w();

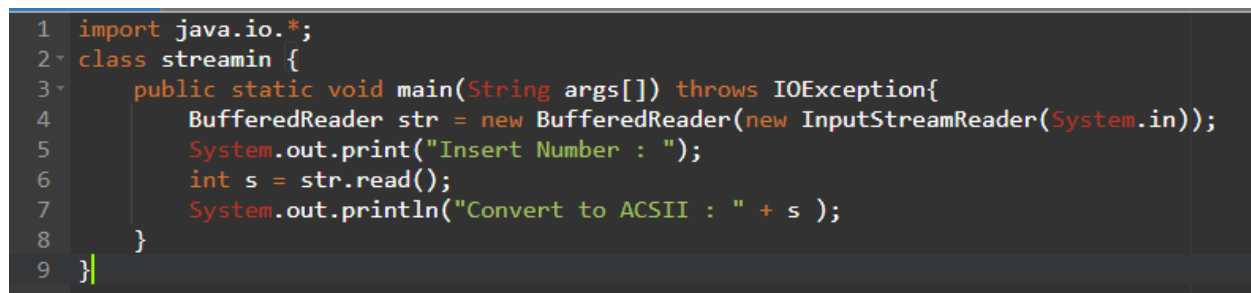
        for (int i=b; i<=c; i++){
            System.out.println(a + "*" + i + "=" + (a*i));
        }
    }
}
```

Command Prompt output:

```
D:\_P\Java\jdk1.8.0_212\bin>javac x5.java
D:\_P\Java\jdk1.8.0_212\bin>java x5
1
2
1*2=2
1*3=3
1*4=4
1*5=5
1*6=6
1*7=7
1*8=8
1*9=9
1*10=10
1*11=11
1*12=12
D:\_P\Java\jdk1.8.0_212\bin>
```

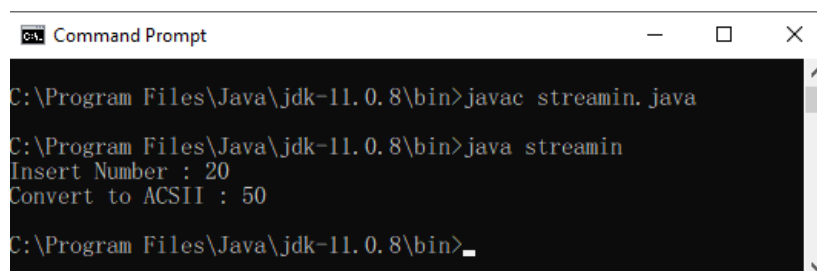
ตัวอย่าง โปรแกรมแม่สูตรคูณ เป็นกรรับค่า argument จาก command line แล้วนำค่าที่ได้ไปประมวลผล

2.การรับค่าจากแป้นพิมพ์ BufferedReader



```
1 import java.io.*;
2 class streamin {
3     public static void main(String args[]) throws IOException{
4         BufferedReader str = new BufferedReader(new InputStreamReader(System.in));
5         System.out.print("Insert Number : ");
6         int s = str.read();
7         System.out.println("Convert to ACSII : " + s );
8     }
9 }
```

ตัวอย่าง การรับค่าจากแป้นพิมพ์ BufferedReader มีการเก็บค่าผลลัพธ์จากแป้นพิมพ์มาอยู่ในตัวแปร s และมีการอ่านค่าผ่าน strผลลัพธ์ คือมีการรับค่าตัวเลขผ่านแป้นพิมพ์เข้ามา และมีการนำตัวเลขที่รับเข้ามาไปเทียบว่าตรงกับตัวเลขตัวที่เท่าไรใน ACSII เสร็จแล้วก็จะประมวลผลออกมา



```
C:\Program Files\Java\jdk-11.0.8\bin>javac streamin.java
C:\Program Files\Java\jdk-11.0.8\bin>java streamin
Insert Number : 20
Convert to ACSII : 50
C:\Program Files\Java\jdk-11.0.8\bin>
```

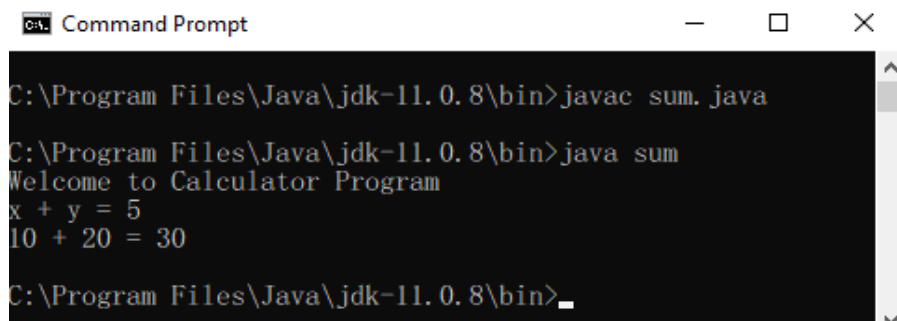
3.การรับค่าเข้า method

```
1 public class sum {
2     public static void main(String[] args) {
3         sayWelcome();
4
5         int x = 2;
6         int y = 3;
7         System.out.println("x + y = " + sum(x, y));
8         System.out.println("10 + 20 = " + sum(10, 20));
9
10    }
11
12    private static void sayWelcome () {
13        System.out.println("Welcome to Calculator Program");
14    }
15
16    private static int sum (int a, int b) {
17        return a + b;
18    }
19 }
```

ตัวอย่าง เราสร้างเมธอดขึ้นมาสองเมธอด เมธอดแรกมีชื่อว่า sayWelcome() เราได้เรียกใช้งานเพื่อให้แสดงข้อความเมื่อเข้าโปรแกรม และเมธอด sum() เป็นเมธอดสำหรับค่าหาผลรวมของตัวเลขสองจำนวน

Info: คำสั่ง static คือคำสั่งที่ทำให้เมธอดหรือตัวแปรใดๆ เป็นตัวแปรที่สามารถใช้ร่วมกันภายในโปรแกรมได้ ในการเรียกใช้งาน static เมธอด เมธอดที่เรียกต้องเป็น static เช่นเดียวกัน ยกเว้นการเรียกใช้แบบ Class prefix ซึ่งคุณจะได้เรียนในบทของออบเจ็ค

การเรียกใช้เมธอดในภาษา Java ใช้ชื่อของมันตามด้วยวงเล็บ () แล้วใส่อาทิวกเมนต์ตามลำดับในลิสต์ของพารามิเตอร์ของเมธอด เราได้เรียกใช้เมธอด sum() สองครั้งโดยไม่ต้องเขียนโปรแกรมการบวกเลขใหม่ ซึ่งนี่เองคือการนำกลับมาใช้ใหม่



```
C:\Program Files\Java\jdk-11.0.8\bin>javac sum. java
C:\Program Files\Java\jdk-11.0.8\bin>java sum
Welcome to Calculator Program
x + y = 5
10 + 20 = 30
C:\Program Files\Java\jdk-11.0.8\bin>_
```

4.การส่งคืนค่าจาก method

```
public class MethodReturn {
    public static void main(String[] args) {

        float pi = getPI();
        System.out.println("Value of PI is " + pi);

        System.out.println("\nDetemine if the number is or not");
        for (int i = 1; i <= 20; i++) {
            if (isPrime(i)) {
                System.out.println(i + " is prime");
            } else {
                System.out.println(i + " is not prime");
            }
        }

    }

    public static float getPI () {
        return 3.14f;
    }

    public static boolean isPrime(int number) {
        for (int i = 2; i < number; i++) {
            if (number % i == 0 && i != number) return false;
        }
        return true;
    }
}
```

ตัวอย่างเป็นการใช้คำสั่ง return เรามีสองเมธอดคือ getPI() ใช้สำหรับรับค่า PI มา ดังนั้นประเภทของฟังก์ชันจะเป็น float เพราะเราต้องการส่งค่าของ floating-point กลับไป

ต่อมาเป็นเมธอด isPrime() เป็นเมธอดสำหรับตรวจสอบตัวเลขจำนวนเฉพาะ โดยค่าที่ส่งกลับเป็น boolean ซึ่งเป็น true ถ้าหากเป็นจำนวนเฉพาะและ false ถ้าไม่เป็น คุณเห็นว่าเราสามารถเรียกใช้เมธอดนี้ถึง 20 ครั้ง ซึ่งเราได้ใช้โค้ดเดิมโดยไม่ต้องเขียนขึ้นใหม่เสมอ

```
Value of PI is 3.14
```

```
Detemine if the number is or not
```

```
1 is prime
2 is prime
3 is prime
4 is not prime
5 is prime
6 is not prime
7 is prime
8 is not prime
9 is not prime
10 is not prime
11 is prime
12 is not prime
13 is prime
14 is not prime
15 is not prime
16 is not prime
17 is prime
18 is not prime
19 is prime
20 is not prime
```

5.การเรียก method จากต่าง class แบบไม่ extends

```
1 // ประกาศ class ด้วย final
2 final class Foo {
3     void x() {
4         System.out.println("X - Foo");
5     }
6 }
7 // ไม่สามารถ extends class ที่ประกาศเป็น final ได้
8 // จะฟ้อง error "Cannot inherit from 'Foo'"
9 class Bar extends Foo {
10 }
```

เมื่อใช้ final ในการประกาศ class จะทำให้ไม่สามารถสืบทอดหรือสร้าง subclass นั้นได้ นั่นคือถ้าประกาศ class ด้วย final แล้วจะทำให้ไม่สามารถ extends class นั้นได้ โดยจะมีการฟ้อง error “Cannot inherit from ‘Foo’”

6.การเรียก method จากต่าง class แบบ extends

```
Car.java
class Vehicle {
    protected String brand = "Ford";
    public void honk() {
        System.out.println("Tuut, tuut!");
    }
}

class Car extends Vehicle {
    private String modelName = "Mustang";
    public static void main(String[] args) {
        Car myFastCar = new Car();
        myFastCar.honk();
        System.out.println(myFastCar.brand + " " + myFastCar.modelName);
    }
}
```

ตัวอย่าง เป็นการเรียก extends มากจากต่าง class โดย class Car ที่ทำการ extends นั้นจะเรียกใช้งาน method ทั้งหมดของคลาส Vehicle ที่เป็น class (parent) ที่ถูก extends มาใช้งาน ผลลัพธ์ : คลาสรถ (คลาสรถย่อย) รับช่วงแอตทริบิวต์และวิธีการจากคลาสรถยนต์ (ซูเปอร์คลาส)

Result:

```
Tuut, tuut!
Ford Mustang
```

7. for แรกรับค่า ส่วน for ที่สองประมวลผล

```
1 import java.io.*;
2 class one {
3     public static void main(String args[]) throws IOException {
4         BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));
5         int i[] = new int[5];
6         int plus = 0, Minus = 0, Multi = 0, Divide = 0;
7         for (int k=0; k<5; k++) {
8             i[k] = Integer.parseInt(stdin.readLine());
9         }
10        for (int k=0; k<5; k++) {
11            plus = i[0] + i[1] + i[2] + i[3] + i[4];
12            Minus = i[0] - i[1] - i[2] - i[3] - i[4];
13            Multi = i[0] * i[1] * i[2] * i[3] * i[4];
14            Divide = i[0] / i[1] / i[2] / i[3] / i[4];
15        }
16        System.out.println("plus = " + plus);
17        System.out.println("Minus = " + Minus);
18        System.out.println("Multi = " + Multi);
19        System.out.println("Divide = " + Divide);
20    }
21 }
```

ตัวอย่าง for รับค่าและ for ประมวลผล เป็นการรับค่าจากแป้นพิมพ์โดยใช้ BufferedReader ในการรับค่ามาเก็บในตัวแปรอาร์เรย์ for loop แรก ใช้ในการรับค่ามี 5 loop ส่วน for loop ที่สองใช้ในการคำนวณค่าที่ loop แรกได้รับค่าเข้ามาจากแป้นพิมพ์

8. ตัวแปรแบบ instance และแบบ local variable

Example:-

```
public class Employee {

    // this is q instance variable, and visible for any ch
    public String empName;

    // empSalary variable is visible in Employee class onl
    private double empSalary;

    // employee name is set in the constructor.
    public Employee (String ename) {
        empName = ename;
    }

    // method to set employee salary.
    public void setSalary(double esalary) {
        empSalary = esalary;
    }
}
```

```
// method prints the employee information.
public void getEmpInfo() {
    System.out.println("Employee Name Is : " + empName);
    System.out.println("Employee Salary Is : " + empSal);
}

public static void main(String args[]) {
    Employee empOne = new Employee("Steve");
    empOne.setSalary(25000);
    empOne.getEmpInfo();
}
}
```

When we run the above java program, we will see the following output.

Output:-

```
Employee Name Is : Steve
Employee Salary Is : 25000.0
```

instance variable : ตัวแปรอินสแตนซ์ถูกประกาศภายในคลาสนอกเมธอดตัวสร้างหรือบล็อกใดๆ ตัวแปรอินสแตนซ์สามารถประกาศได้ในระดับคลาสนี้และสามารถมองเห็นได้สำหรับเมธอดตัวสร้างและบล็อกทั้งหมดในคลาสนี้ ตัวแปรอินสแตนซ์มีค่าเริ่มต้น ค่าสำหรับตัวแปรอินสแตนซ์สามารถกำหนดได้ในระหว่างการประกาศหรือภายในตัวสร้าง ตัวแปรอินสแตนซ์สามารถเข้าถึงได้โดยตรงโดยการเรียกชื่อตัวแปรภายในคลาสนี้ แต่สำหรับวิธีการแบบคงที่ควรเรียกโดยใช้ชื่อแบบเต็มตัวแปร Instance ถูกสร้างขึ้นเมื่ออ็อบเจกต์ถูกสร้างและทำลายเมื่ออ็อบเจกต์ถูกทำลาย พวกเขายังเรียกว่าฟิลด์ ตัวระบุการเข้าถึงสามารถกำหนดให้กับตัวแปรอินสแตนซ์และหากไม่มีการกล่าวถึงจะใช้ตัวระบุเริ่มต้น

Example:-

```
public class Age {
    public void getAge() {
        int age = 25;
        System.out.println("My age is : " + age);
    }

    public static void main(String args[]) {
        Age test = new Age();
        test.getAge();
    }
}
```

Here, **age** is a local variable. This is defined inside **getAge()** method and its scope is limited to only this method. When we run the above java program, we will see following output.

Output:-

```
My age is: 25
```

local variable : ตัวแปรที่ประกาศภายในเมธอดตัวสร้างหรือบล็อกโค้ดเรียกว่าตัวแปรโลคัล สามารถเข้าถึงได้เฉพาะในวิธีการหรือบล็อกโค้ดนั้นเท่านั้น ตัวแปรภายในไม่พร้อมใช้งานนอกฟังก์ชันที่กำหนดไว้บล็อกเริ่มต้นด้วยวงเล็บปีกกาเปิดและลงท้ายด้วยวงเล็บปีกกาปิด ไม่สามารถใช้ตัวระบุการเข้าถึงสำหรับตัวแปรโลคัล

9.การทำ casting

```
class Main {  
    public static void main(String[] args) {  
        // create string type variable  
        String data = "10";  
        System.out.println("The string value is: " + data);  
  
        // convert string variable to int  
        int num = Integer.parseInt(data);  
        System.out.println("The integer value is: " + num);  
    }  
}
```

ตัวอย่าง การแปลงชนิดข้อมูลจากชนิดหนึ่งไปเป็นอีกชนิดหนึ่ง ผลลัพธ์แปลง String เป็น int

```
The string value is: 10  
The integer value is: 10
```