I am assuming you know at least one programming language and the concept of object / pointers. I will mention algorithms and data structures in increasing order of difficulty.

First start with Linear data strucures and algorithms.

- **Arrays**
- Linked List
- Stack
- Queues
  Then move to basic algorithms :

- Sorting - Merge Sort, Insertion Sort, Quick Sort, Number of inversions
- Matrix Multiplication (just know the algo if not implement it)
- Prime Sieving
- Modular Math including multiplication and division
- Euclidean Algorithm for GCD, Modular Inverse, Fast Exponentiation
- Fibonacci number with matrix multiplication
- Probability distribution and expected value
- Stats - Mean, Median, Variance, Bayes theorem
  The one can learn some popular algorithmic techniques:

- Divide and Conquer - Binary Search, Maximum Subarray
- Greedy Algorithms - Activity Selection, Huffman encoding
- Dynamic Programming - Matrix Chain Multiplication, Knapsack,
- Linear Programming - Variable Maximisation, Linear time sorting
- String Algorithms - Manacher, LCS, Edit Distance
  Then comes some typical data structures:

- Trees - Binary Tree, General Tree, Lowest Common Ancestor
- Binary Search Tree - Inorder Traversal, Level order traversal, finding kth largest element, diameter, depth, number of nodes, etc.
- Heaps - Array Implementation, Heapify, Heap Sort
- Union Find
- Hash Table - Linear Probing, Open addressing, Collision avoidance
  Then you can learn about Graphs:

- Adjacency List, Adjacency Matrix, Weighted Edge Graphs
- Basic Traversal algos - Breadth First Search, Depth First Search, etc
- Shortest Path Finding Algorithm - Dijkstra, Floyd Warshal, Bellman Ford
- Minimum Spanning Tree - Kruskal's Algo, Prim's Algo
  By this time you are already pretty good with programming. You will do better than most of undergrad CS students. If you want to learn more and delve deep read more.

Advance Tree and Graph :

- Balanced Trees - AVL, Red-Black
- Heavy Light Decomposition, B+ Trees, Quad Tree
- Advance Graph - Min Cut, Max Flow
- Maximum Matching - Hall's Marriage
- Hamiltonian Cycle
- Edge Graphs / Line Graphs
- Strongly Connected Components
- Dominant Sub-Graph, Vertex Cover, Travelling Salesman - Approx algos

Advance String Algorithms :

- Knuth Morris Pratt Algorithm
- Rabin Karp Algorithm
- Tries and Compressed Tries
- Prefix Trees, Suffix Trees, Suffix Automation - Ukkonen Algorithm

Advance Math:

- Fast Fourier Tranformation
- Primality Testing
- Computational Geometry - Closest point pair, Voronoi diagram, Convex Hull

General Advance topics :

- Iterating through all combination / permutation
- Bit manipulation

Java Program:

1. **Program to print "Hello"**
2. **Program to add two number.**
3. **Program to add two number by user.**
4. **Program to swap two number.**
5. **Program of simple interest**
6. **Program of array**
7. **Program of Fibonacci series**
8. **Program of factorial**
9. **Program of greatest among the number**
10. **Program of ternary operator**
11. **Program of odd even number**
12. **Program of Palindrome number**
13. **Program of Program of position**
14. **Program of Prime number**
15. **Program of Room Area**

16. **Program of sorting**
17. **Program of table**
18. **Program of implicit type casting**
19. **Program of explicit type casting**

Why Java:

We have the window operating system, we have Linux, Apple and the list goes on for different operating system. The number of operating system in the world are quite high. If you are an engineer and your job is to make the application and the application, software etc but your customer is using different kind of operating system around the world. It's very hard and time taken for the engineer to develop the application for different user.

If you have developed software for windows in 90 days, then you have to do same work for apple, Linux etc for another 90+90 Days.

After this hilarious problem, the engineer of the Silicon Valley came up with an Idea called JVM (Java virtual Machine). From this the Engineer all need to code for JVM in Java Then JVM will guarantee that your application will be run either on Linux, Apple window etc. You no need to code for different machine and learn different language.

Java code is compiled by the compiler and converted into byte code. This byte code is platform independent code because it can be run on multiple platform either on Linux, windows or Apple. You have to write the code only once but the JVM run it anywhere.

Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, Mac/OS etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere(WORA).

With the help of JVM you all need to code for the JVM in Java programming Language then JVM will translate to machine code for the different machine.JVM is nothing but a byte code which is run on the machine not exist

So if you want to be programmer then learn and code in Java programming language not the different kind of the languages.

public static void main(String args[])

Like c/c++ we used the main method in java but we use the different keyword in main function of the Java and all of these have different meaning. Every java program used the class and every class have the main function which is the entry point of the code execution.

1. **Public:** Public is an access specifier. An access specifier can be private, protected or public which define the data in which way you can access the class. Since main function

is always called outside the class by the JVM so we have to define the main class public at any instant.

2. **Static:** Static is the keyword in java and it can be applicable to the Method Variable, Class nested within another Class, Initialization Block etc but can't implemented to the constructor, interfaces, non-nested class, local variables etc. By the help of static in public static void main(String args[]) we indicates that this method can be called without creating the object of the class. If you don't use the static keyword with the main function, then you have to instantiate the class and call this method from the resulting object.
   We cannot declare the main method without using the static keyword because everything is act as an object and everything is invoked by the object Only static keyword is the way to run the main method without creating the object because it create the memory location in the context area which help the program for execution. Without the main method to be declared as the static you can compile the program successfully but your program will not run.
3. **Void:** we all know that void means empty. So the use of void in public static void main(String args[]) indicates that this method doesn't return anything or return no value. Since the JVM is the function, so it is supposed to return a value to the caller i.e. JVM. So by placing the void before the main function we assure to main function is returning nothing.
4. **Main:** main is the function name. It is special name to start the execution of the program. Every java program is start with the main method.
5. **String [] args:** It is the array of the string with name args. You can use any of the name instead the args. This string array is used to access the command line arguments. This string store the character string and the argument received the command-line arguments present when the program is executed .

Scanner S=new Scanner(System.in);

There is various way to read input from the keyboard in Java.For example **java.util.Scanner;**

1. In **Scanner S=new Scanner(System.in);** Scanner is the class which is define in the java.util.Scanner; It is used to read the input from keyboard or network in the Java.
2. **S** means we are going to create a new object of the class Scanner.
3. The System.in tell the compiler to read the data typed in the keyboard.

System.out.println("…");

System.out.println("…"); is most used java function in the history but most of the programmer unable to explain this function.It is so important and have asked in many MNC interviews or in college viva.

1. **System:** System is the class name which is present in the java.lang packakge.
2. **out** is a static variable present in system class of type PeintStream.
3. **Println** is a method present in PrintStream class

What is JDK , JVM and JRE

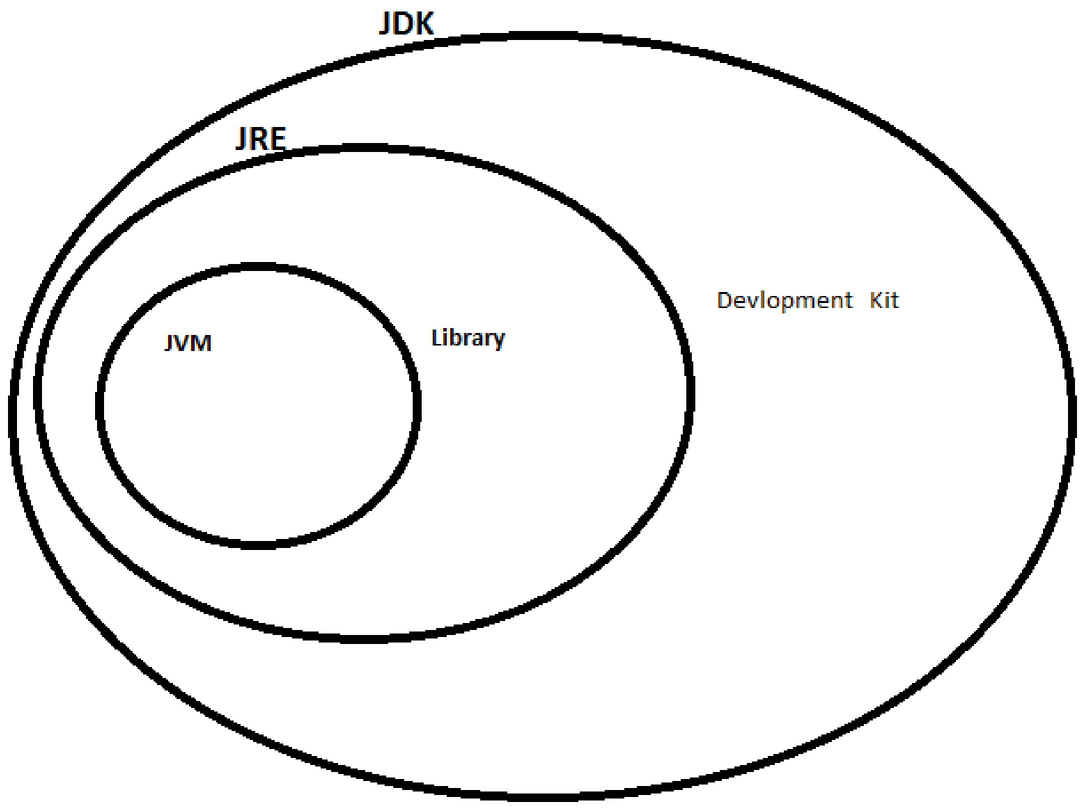
JDK:

JDK is stand for the Java development kit which is responsible for the development and execution of the Java program. It provides the environment to develop, compile, debug and run a Java program consist of compiler, application launcher and debugger. Compiler convert the Java code into machine instruction, Java application launcher start the application and the debugger helps to debug a program. JDK is platform dependent software so we have separate installation of it for windows MAC and Unix.

JRE:

JRE is stand for the Java Runtime Environment and it is need for the execution of the JAVA program. If you want to run a java program, then you have to installed the JRE. It contains the JVM which is Java virtual machine and the and the library functions. JVM is also platform dependent software and it uses the class library and run the java byte code. The task of the is to Loads code, verifies code, Executes code and Provides runtime environment.

JVM is an abstract machine while on the other hand JRE is the acronym.



So,

JDK=JRE + Devlopment kit

JRE=JVM + library

Type casting: To change the variable of one type into another type

There are two types of type casting

1. Implicit Type casting
2. Explicit Type casting

1. Implicit type casting:

// Demonstrate the following program

class Saiful

{

public static void main(String[] args)

{

int x='a';

System.out.println(x);

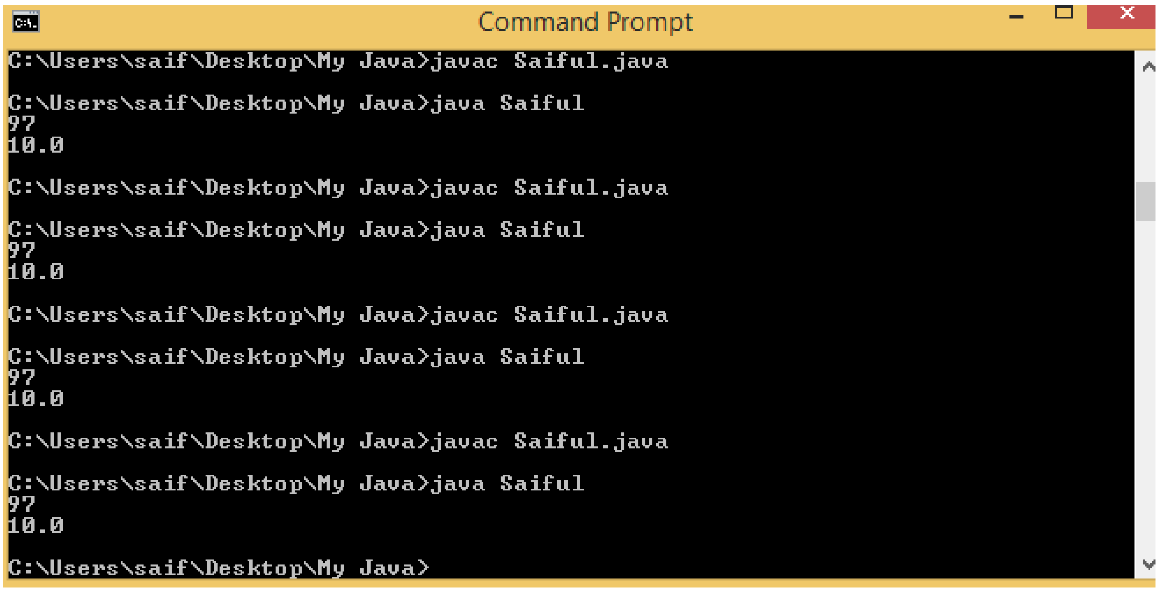
double d=10;

System.out.println(d);

}

}

Output:



```
C:\Users\saif\Desktop\My Java>javac Saiful.java

C:\Users\saif\Desktop\My Java>java Saiful
97
10.0

C:\Users\saif\Desktop\My Java>javac Saiful.java

C:\Users\saif\Desktop\My Java>java Saiful
97
10.0

C:\Users\saif\Desktop\My Java>javac Saiful.java

C:\Users\saif\Desktop\My Java>java Saiful
97
10.0

C:\Users\saif\Desktop\My Java>javac Saiful.java

C:\Users\saif\Desktop\My Java>java Saiful
97
10.0

C:\Users\saif\Desktop\My Java>
```

In this program the character a is print as a integer value 97 and the integer value 10 is print as a double value 10.0

Internally compiler converts the character value into integer value and integer value into double value by implicit type casting.

In implicit type casting smaller data type value is assigning to the bigger data type variable. In the program character a is 2 byte which is converted to the integer 10 which is 4 byte in the similar war integer 10 is 4 byte and it is converted to the double 10.0 which is 8 byte. It is also known as the widening or up casting

2. **Explicit type casting:**

// consider the following program

class Explicit

{

public static void main(String[] args)

{

int a=10;

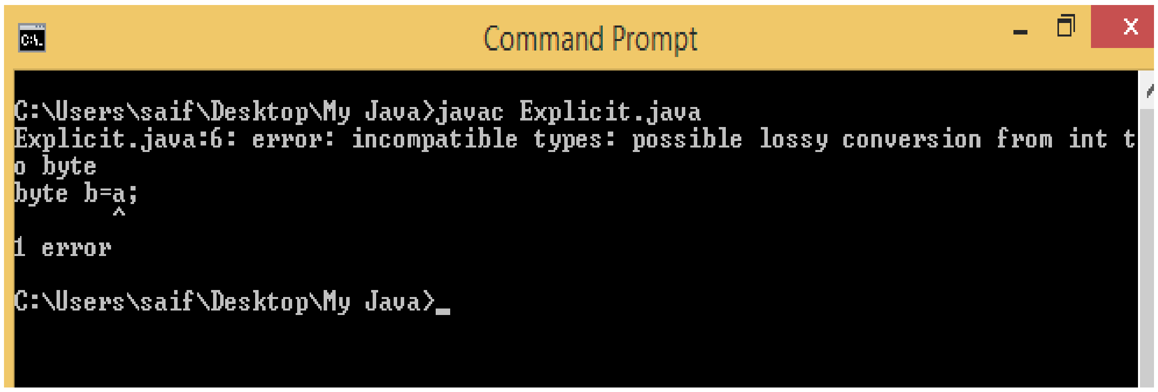
byte b=a;

System.out.println(b);

}

}

If you run this program then compiler will not execute the program because there is loss of information because bigger data type value is converted into smaller data type variable.

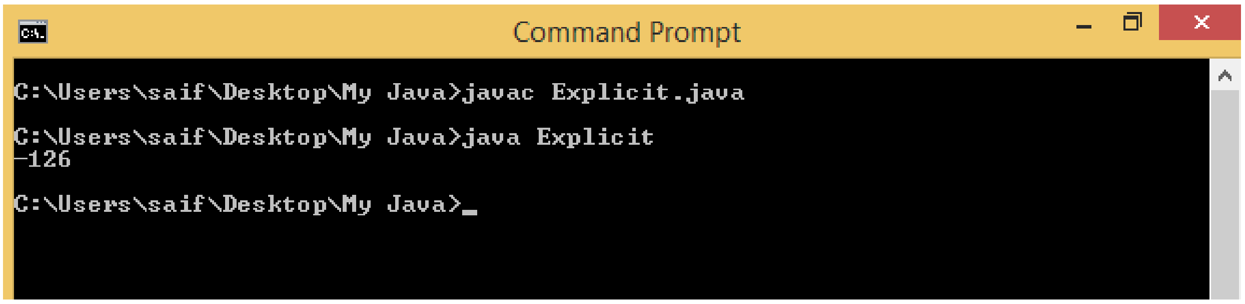


```
C:\Users\saif\Desktop\My Java>javac Explicit.java
Explicit.java:6: error: incompatible types: possible lossy conversion from int t
o byte
byte b=a;
       ^
1 error

C:\Users\saif\Desktop\My Java>_
```

But if write the program in this way

```
class Explicit
{
public static void main(String[] args)
{
int a=130;
byte b=(byte)a;
System.out.println(b);
}
}
```

Then you told the complier to type cast the value without worry about the loss of information the compiler will execute the program and cast the value

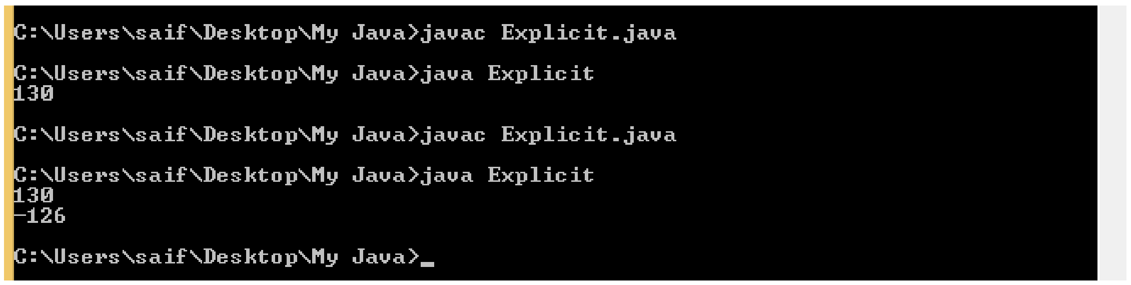


```
C:\Users\saif\Desktop\My Java>javac Explicit.java

C:\Users\saif\Desktop\My Java>java Explicit
-126

C:\Users\saif\Desktop\My Java>_
```

The programmer is responsible for the explicit type casting and the bigger data type value is converted into smaller data type variable. It is also called the narrowing and down casting and there is loss of information in down type casting.

Knowledge+1: Program in java on explicit type casting

```
class Explicit
{
public static void main(String[] args)
{
int a=130;
short b=(short)a;
System.out.println(b);
int a1=130;

byte b1=(byte)a1;

System.out.println(b1);

}

}
```



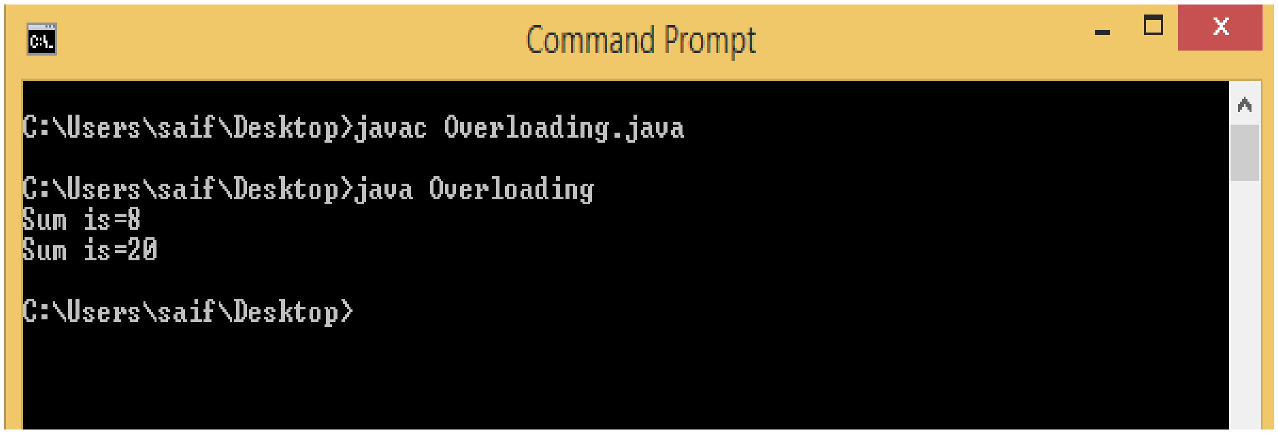
```
C:\Users\saif\Desktop\My Java>javac Explicit.java

C:\Users\saif\Desktop\My Java>java Explicit
130

C:\Users\saif\Desktop\My Java>javac Explicit.java

C:\Users\saif\Desktop\My Java>java Explicit
130
-126

C:\Users\saif\Desktop\My Java>_
```

Overloading:
In method overloading two or more function have the same name but different argument. The argument may be differing in terms of type or number of arguments invoked in the function is different.Method overloading is compile time polymorphism in which JVM understand at compile time which method should call based on the number or type of arguments call by the compiler.
Note: In java method overloading function should have same return data type.Method overloading function is not possible for different return data type.
Use of the method overloading:
The method overloading increase the readability of the program to the developer and it also save the memory at compile time.
In method Overloading:
1. Inheritance is not allowed
2. Method overloading are always occur in the same class.
3. One overload method does not hide another overload method.
4. In method overloading parameter must be different to each other.
5. In method overloading return type may or may not be different
6. Jave also does not support overloading based on the return data type.

Program of Method overloading when parameter is different:

```
import java.util.Scanner;
class Overloading
{
void sum(int x, int y)
{
System.out.println("Sum is="+(x+y));
}
void sum(int x, int y,int z)
{
System.out.println("Sum is="+(x+y+z));
}
public static void main(String[] args)
{
Overloading o=new Overloading();
o.sum(5,3);
o.sum(8,9,3);
}
}
```

Output:

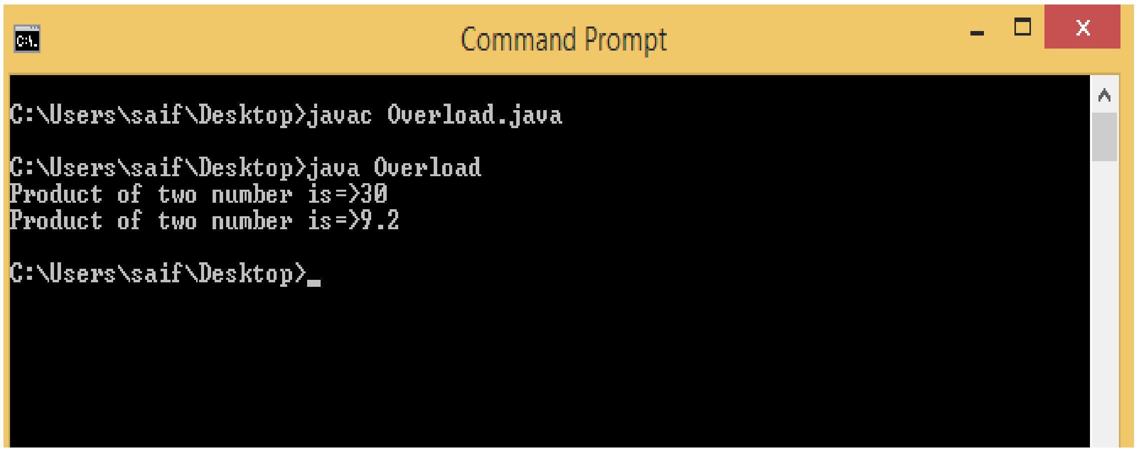


```
C:\Users\saif\Desktop>javac Overloading.java

C:\Users\saif\Desktop>java Overloading
Sum is=8
Sum is=20

C:\Users\saif\Desktop>
```

Program of Method overloading when parameter type is different:

```
import java.util.*;
class Overload
{
void mul(int x,int y)
{
int s=x*y;
System.out.println("Product of two number is=>"+s);
}
void mul(int x,double y)
{
double s=x*y;
System.out.println("Product of two number is=>"+s);
}
public static void main(String[] args)
{
Overload o=new Overload();
o.mul(5,6);
o.mul(2,4.6);
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Overload.java

C:\Users\saif\Desktop>java Overload
Product of two number is=>30
Product of two number is=>9.2

C:\Users\saif\Desktop>_
```

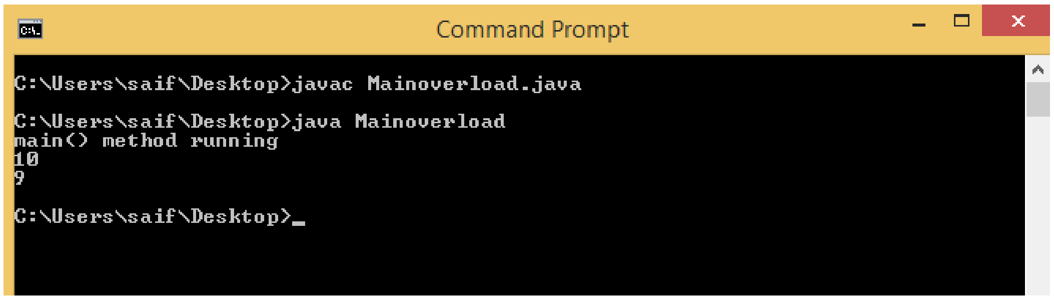
Q. Can we overload the main method?

Yes we can overload the main method. But at starting point the compiler enter into the code if there is main method with string args. If there is no method only with string argument then complier will throw the error and exit from the code. To overload the main method we need to create different main method except the main method with string args. All the other method will treat as a normal method.

Program in Java for main Method overloading

```
class Mainoverload
{
  public static void main(int a)
  {
  System.out.println(a);
  }
public static void main(int a,int b)
  {
  System.out.println(a+b);
  }
  public static void main(String args[]){
  System.out.println("main() method running");
  main(10);
  main(4,5);
  }
}
```

Output:



```
C:\Users\saif\Desktop>javac Mainoverload.java

C:\Users\saif\Desktop>java Mainoverload
main() method running
10
9

C:\Users\saif\Desktop>_
```

Constructor:
Constructor is a type of method which has the same the same name as class name and it does not have any return data type even we cannot return void.It's main function is to initialize the value to the object.

Properties of the constructor:

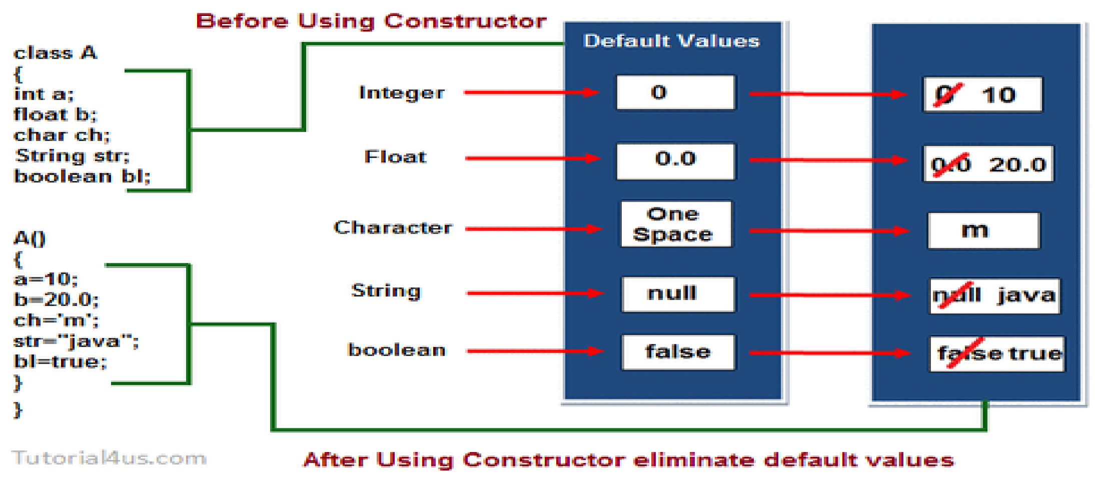
1. Constructor name is always same as the class name.
2. It is used for initializing member element or set the initial set of object.
3. Constructor always be the call when the object is created.
4. Constructor does not have any return type value even the void.
5. Constructor are always invoked at the time of object creation.
6. Constructor are not called like other method function.It is the part of the process of object creation.
7. Apart from initialization constructor also perform the following task like object creation , starting a thread and calling method.

Constructor Example with real world scenario:

1. We know that the we make the blue print of house in order to mentions the length width and type of the internal and external structure of the house. I the same way, a class is created first, that mentions the variables, methods of the class, constructor, instance block and the static block.
2. So, blue print is not the house itself. Similarly, class is not an object itself
3. So if you want to construct the house ob the base of blue print, then you need to allocate the space or area for it. Similarly, in order to create an object of class you need to allocate memory for it (to store all its variables)

So, Constructor is a way to build an object.

How constructor initialize an object:



When we say, MyClass objC = new MyClass();

When the run time that executes this statement, it follows the blueprint (that is the class definition) and allocates the memory for actual instance of MyClass. So when you assign some values to the variables in objC, it actually gets stored in the memory space that was allocated to it.

Constructors are of two types:

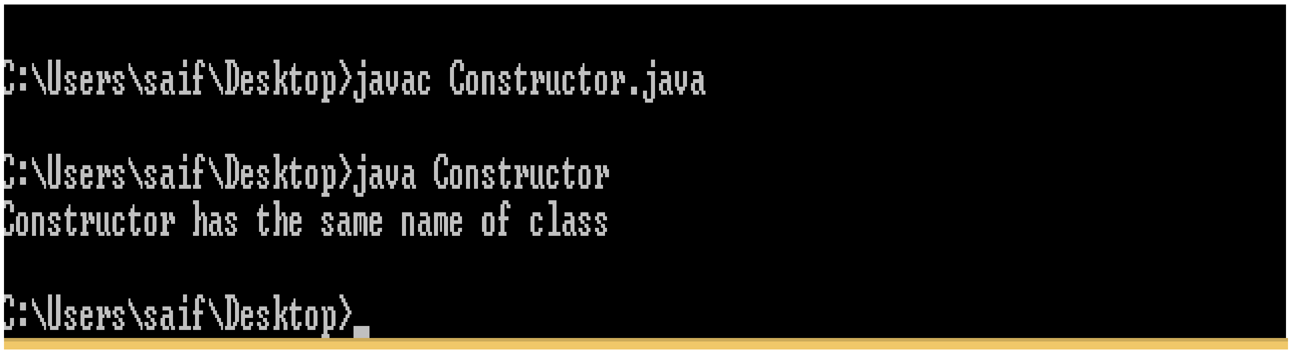
1. Default constructor or no argument construtor
2. Parameterized constructor

Default constructor provides the default value to the object like zero null etc.

Program of default constructor:

```
import java.util.*;
class Constructor
{
Constructor()
{
System.out.println("Constructor has the same name of class");
}
public static void main(String[] args)
{
Constructor c=new Constructor();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Constructor.java

C:\Users\saif\Desktop>java Constructor
Constructor has the same name of class

C:\Users\saif\Desktop>_
```

Note: If there is no constructor in the class then compiler will automatically create the default constructor.

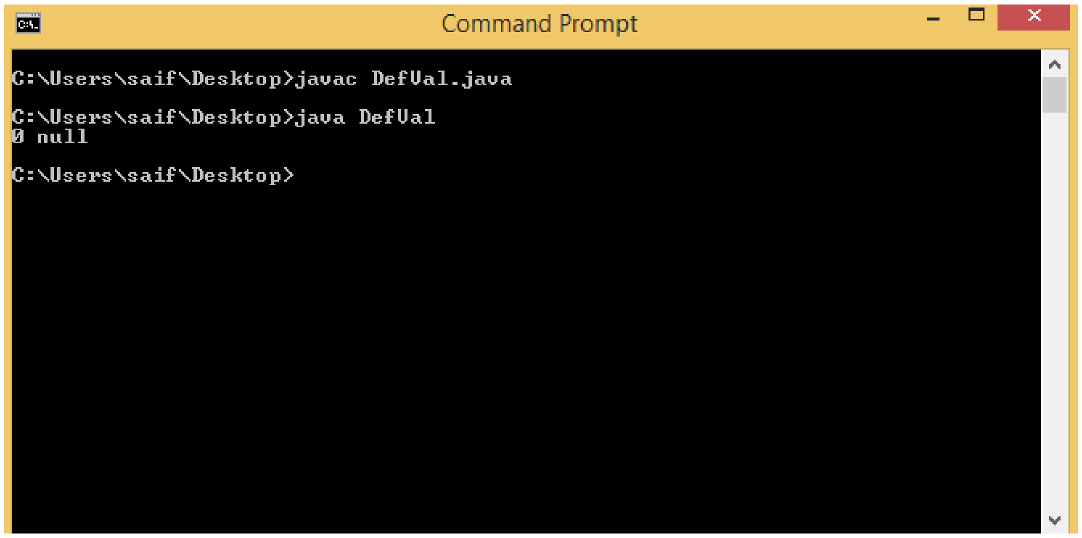
Default constructor provides the default value to the object like zero null etc.

For example:

```
class DefVal
{
int i;
String name;
void display()
{
System.out.println(i+" "+name);
}
public static void main(String[] args)
{
DefVal o=new DefVal();
o.display();
}
```

}

Output:



```
C:\Users\saif\Desktop>javac DefVal.java

C:\Users\saif\Desktop>java DefVal
0 null

C:\Users\saif\Desktop>
```

In the above case we are not creating any constructor while the default constructor is created by the compiler and return the default value.

Difference between constructor and function in java

Constructor	Method
1. Must have the same name as that of the class	May or may not have the same name of the class
2. It is used to initialize the object	It is a set of code to perform specific operation
3. Constructor cannot have the return data type even the void	Function must have the return data type
4. There is the default constructor provided by the java compiler	There is no default constructor provided by the java.
5. It is an implicit method	It is an explicit method

How to copy value from one constructor to another:

class Copy

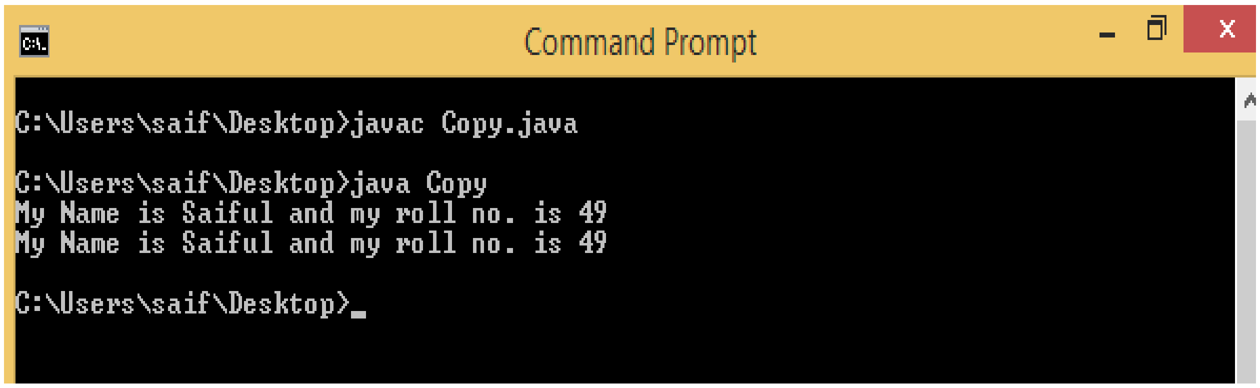
{

int roll;

String name;

```
Copy(int i, String n)
{
roll=i;
name=n;
}
Copy()
{
}
void Display()
{
System.out.println("My Name is "+name+" and my roll no. is "+roll);
}
public static void main(String[] args)
{
Copy c1=new Copy(49,"Saiful");
Copy c2=new Copy();
c2.roll=c1.roll;
c2.name=c1.name;
c1.Display();
c2.Display();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Copy.java

C:\Users\saif\Desktop>java Copy
My Name is Saiful and my roll no. is 49
My Name is Saiful and my roll no. is 49

C:\Users\saif\Desktop>_
```

Static Keyword:

Static is a keyword in java with the member function and member variable of the class so that you call those member without using the object or without creating the object. You don't instantiate the class to call that member.

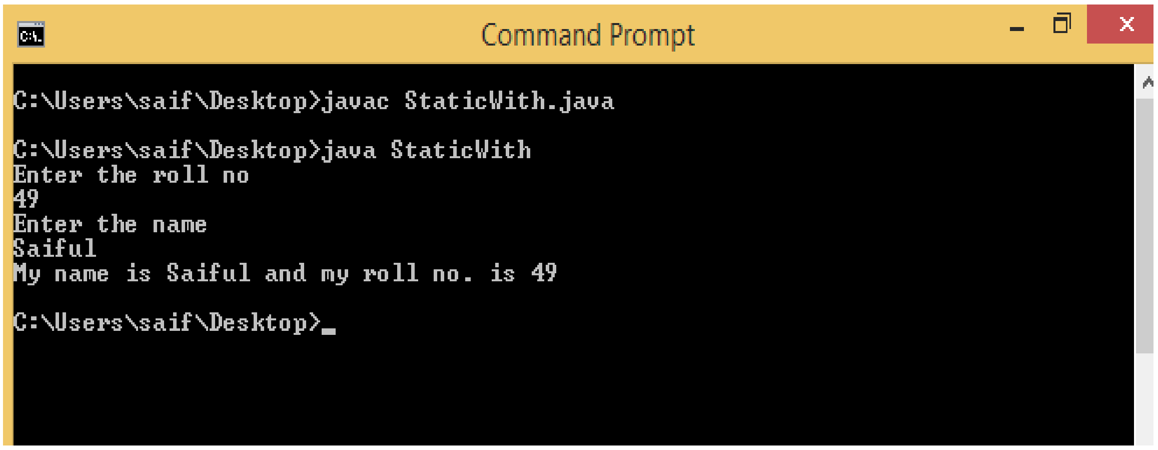
Properties of the static keyword in Java

1. Static keyword in Java is mainly used for the memory management.
2. A static keyword allocates the memory only once at the time of class loading.
3. It makes your program memory efficient and it saves the memory.

Program without static keyword

```
import java.util.*;
class StaticWith
{
int roll;
String name;
StaticWith(int i, String n)
{
roll=i;
name=n;
}
public  void Display()
{
System.out.println("My name is"+name+" and my roll no. is "+roll);
}
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the roll no");
int i=in.nextInt();
System.out.println("Enter the name");
String n=in.next();
StaticWith sw=new StaticWith(i,n);
sw.Display();
}
}
```

Output:

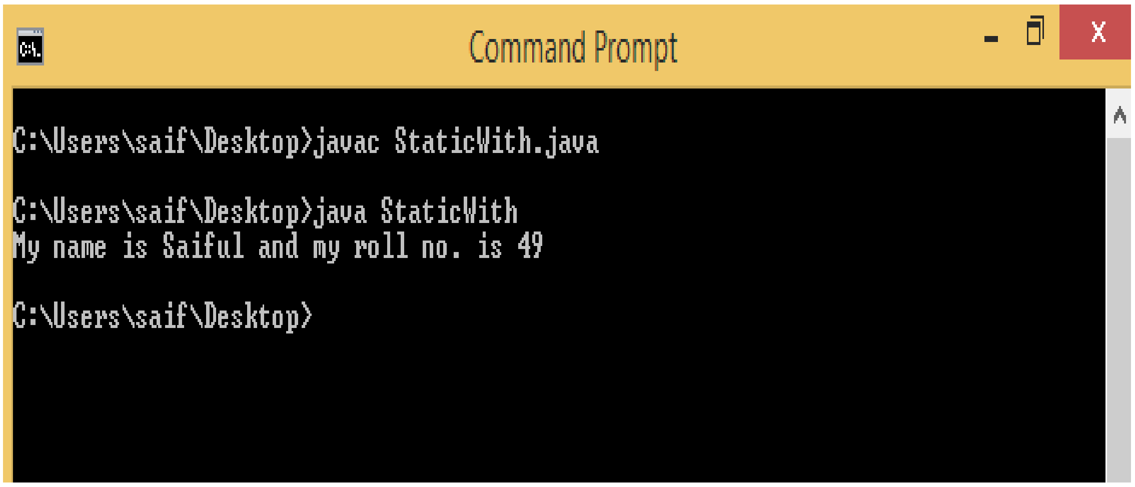


```
C:\Users\saif\Desktop>javac StaticWith.java

C:\Users\saif\Desktop>java StaticWith
Enter the roll no
49
Enter the name
Saiful
My name is Saiful and my roll no. is 49

C:\Users\saif\Desktop>_
```

Program of Static keyword help in no need to create the object and call it by the class name.

```
import java.util.*;
class StaticWith
{
static int roll=49;
static String name="Saiful";
public static void Display()
{
System.out.println("My name is "+name+  " and my roll no. is "+roll );
}
public static void main(String[] args)
{
StaticWith.Display();
}
}
```

Output:



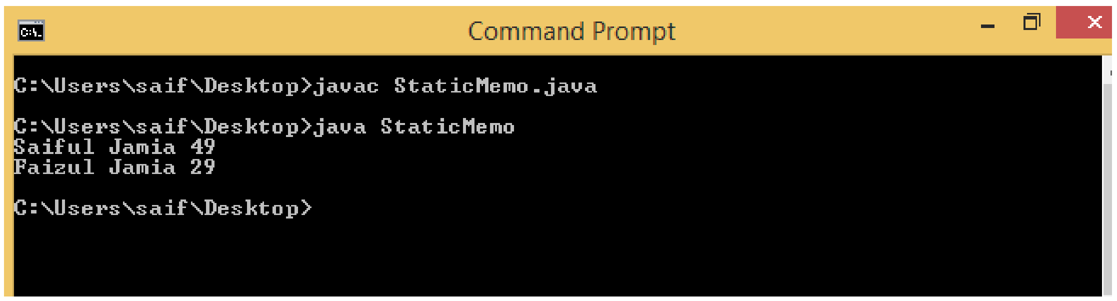
```
C:\Users\saif\Desktop>javac StaticWith.java

C:\Users\saif\Desktop>java StaticWith
My name is Saiful and my roll no. is 49

C:\Users\saif\Desktop>
```

How static variable allocates memory one time:

```
class StaticMemo
{
int roll;;
String name;
static String college="Jamia";
StaticMemo(int i, String n)
{
```

```
roll=i;
name=n;
}
public void Display()
{
System.out.println(name+" "+college+" "+roll);
}
public static void main(String[] args)
{
StaticMemo sm1=new StaticMemo(49,"Saiful");
StaticMemo sm2=new StaticMemo(29,"Faizul");
sm1.Display();
sm2.Display();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac StaticMemo.java

C:\Users\saif\Desktop>java StaticMemo
Saiful Jamia 49
Faizul Jamia 29

C:\Users\saif\Desktop>
```

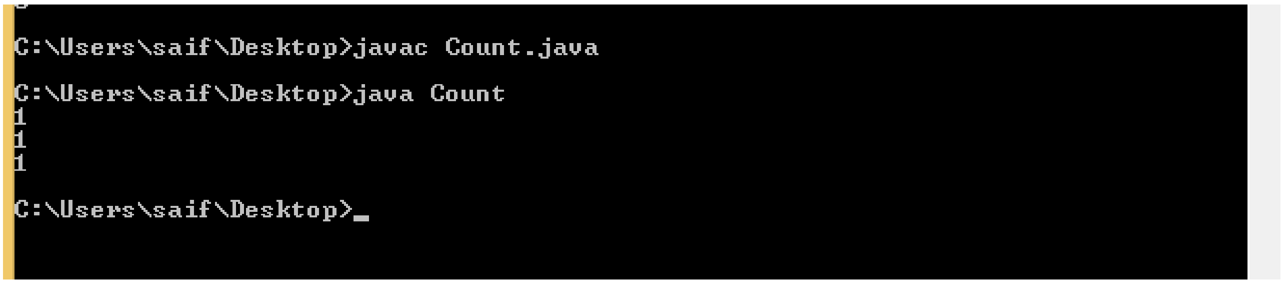
Since college name is common to both so we allocates its memory only once for separate student student with the help of static keyword.

How static keyword retain its memory:

=>Program without static keyword:

```
class Count
{
```

```
int c=0;

Count()

{

c++;

System.out.println(c);

}

public static void main(String[] args)

{

Count c1=new Count();

Count c2=new Count();

Count c3=new Count();

}

}
```



```
C:\Users\saif\Desktop>javac Count.java

C:\Users\saif\Desktop>java Count
1
1
1

C:\Users\saif\Desktop>_
```

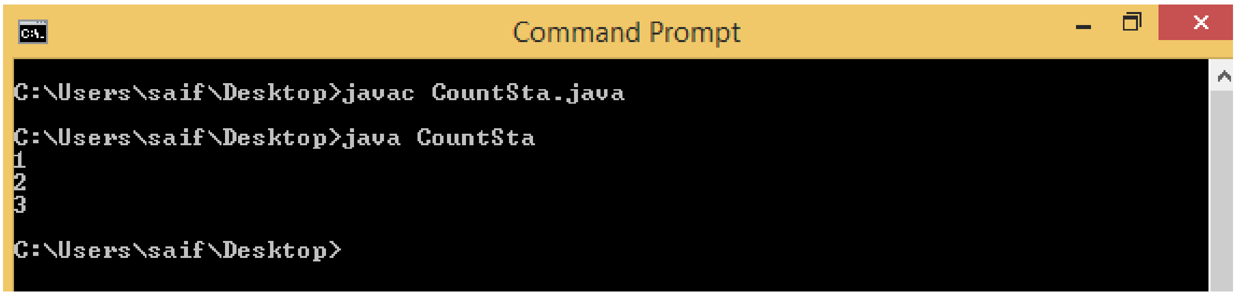
Value remain constant since Since instance variable gets the memory at the time of object creation, each object will have the copy of the instance variable, if it is incremented, it won't reflect to other objects. So each objects will have the value 1 in the count variable.

=>**Program with static keyword:**

```
class CountSta

{

static int c=0;

CountSta()

{
```

```
c++;
System.out.println(c);
}
public static void main(String[] args)
{
CountSta c1=new CountSta();
CountSta c2=new CountSta();
CountSta c3=new CountSta();
}
}
```

Output:

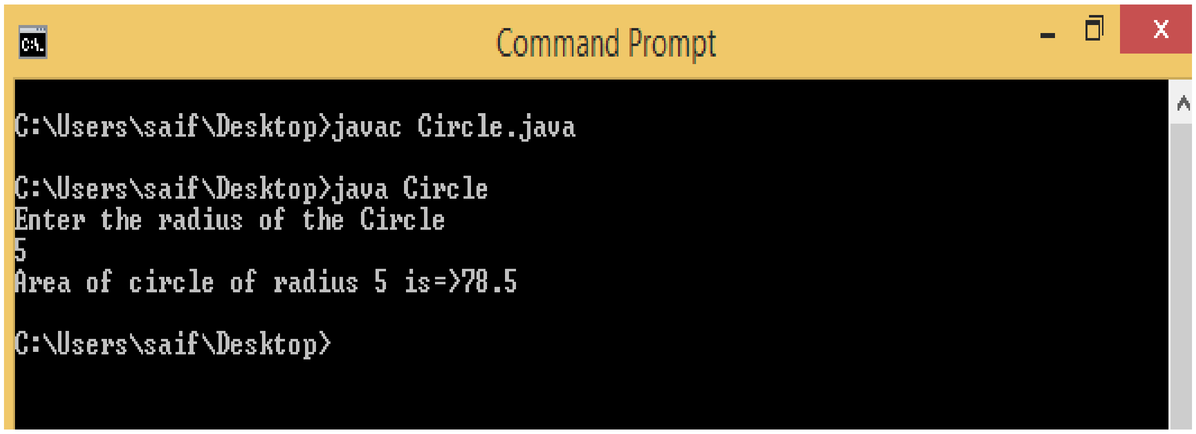


```
C:\Users\saif\Desktop>javac CountSta.java

C:\Users\saif\Desktop>java CountSta
1
2
3

C:\Users\saif\Desktop>
```

Here static variable retain its value..

Static method program in Java:

```
import java.util.*;
class Circle
{
static int x;
void Area(int i)
{
x=i;
System.out.println("Area of circle of radius "+x+" is=>"+3.14*x*x);
}
public static void main(String[] args)
{
Circle c=new Circle();
Scanner in=new Scanner(System.in);
System.out.println("Enter the radius of the Circle");
int i=in.nextInt();
c.Area(i);
}
}
```



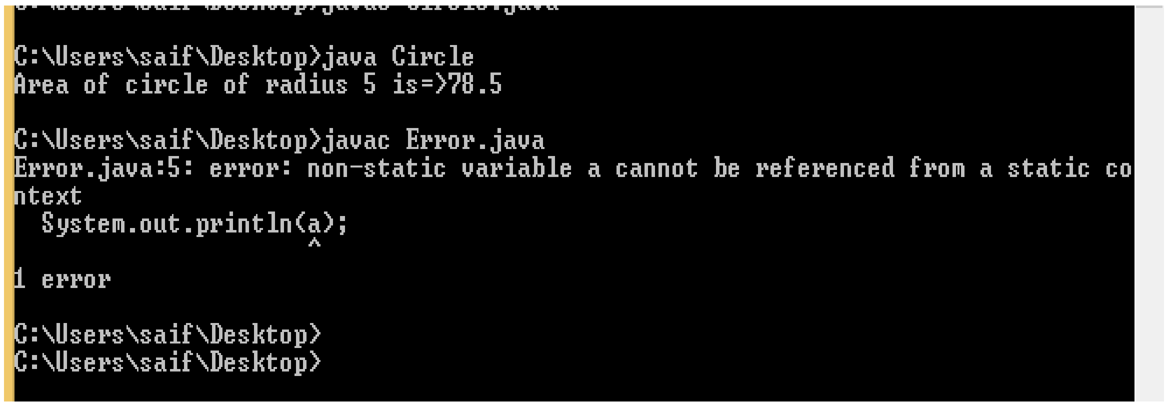
```
C:\Users\saif\Desktop>javac Circle.java

C:\Users\saif\Desktop>java Circle
Enter the radius of the Circle
5
Area of circle of radius 5 is=>78.5

C:\Users\saif\Desktop>
```

Error: In the program a very common problem arises that non static method can not be referenced from the static method.

```
class Error{
int a=5;//non static
public static void main(String args[]){
System.out.println(a);
}
}
```

In the above program, int x is a non-static variable and we call it from the main method which is static. When we want to execute this program compiler will give a compile time error as:

Output Error:



```
C:\Users\saif\Desktop>java Circle
Area of circle of radius 5 is=>78.5

C:\Users\saif\Desktop>javac Error.java
Error.java:5: error: non-static variable a cannot be referenced from a static co
ntext
  System.out.println(a);
                     ^

1 error

C:\Users\saif\Desktop>
C:\Users\saif\Desktop>
```

Static block:

Static block are the block which are executed when your class are loaded in the JVM.

At the run time two action are performed. 1st action is JVM load corresponding .class byte code into the memory and the 2nd action is main method is executed inside the memory.

So when the .class byte code loaded into the memory, then the static block is executed.

Since . class loaded inside the memory once time in the similar way Static block also executed only once time into the memory.

Properties of the static block:

1. They are loaded when the .class byte code is loaded by the JVM inside the memory.
2. Since .class loaded only one time so static block are also executed only one time.
3. We cannot put the static block inside any method.
4. **From** the static method you cannot refer to any instance variable.
5. You can have as many static block as you want and the order in which they executed depend on the placing of static block in the source code.

Why static block is not put inside any method?

There is no point to put the static block inside the method because static block always execute before the main method.

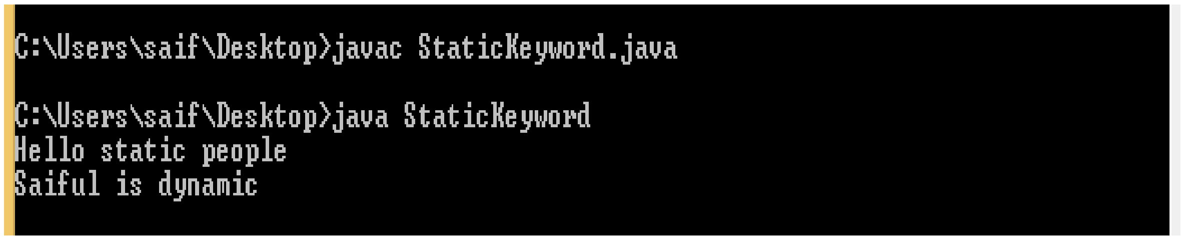
The order in which JVM execute the task is =>

- Static data members (if any) >Static blocks (if defined) >The task that results in class loading is performed>Main method.

Program: Static Keyword Program:

```
class StaticKeyword

{

static

{

System.out.println("Hello static people");

}

public static void main(String[] args)

{

System.out.println("Saiful is dynamic");

}
```

Output:



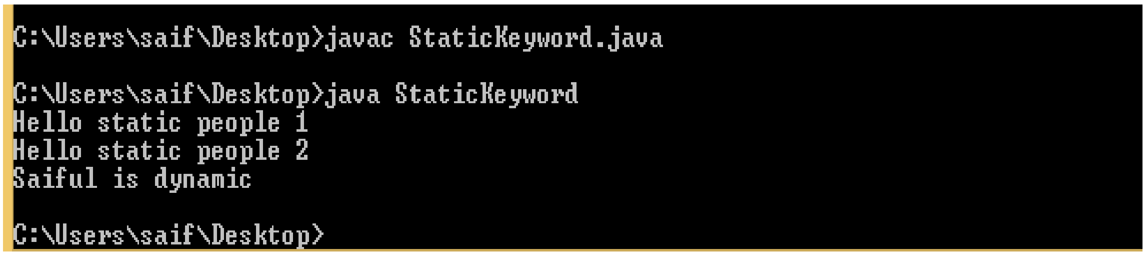
```
C:\Users\saif\Desktop>javac StaticKeyword.java

C:\Users\saif\Desktop>java StaticKeyword
Hello static people
Saiful is dynamic
```

Program: Order of static block and when main method is executed:

```
class StaticKeyword

{

static

{

System.out.println("Hello static people 1");

}

public static void main(String[] args)

{

System.out.println("Saiful is dynamic");

}

static

{
```

```
System.out.println("Hello static people 2");
}
}
```

Output:



```
C:\Users\saif\Desktop>javac StaticKeyword.java

C:\Users\saif\Desktop>java StaticKeyword
Hello static people 1
Hello static people 2
Saiful is dynamic

C:\Users\saif\Desktop>
```

this keyword in java:

Demonstrate the following program:

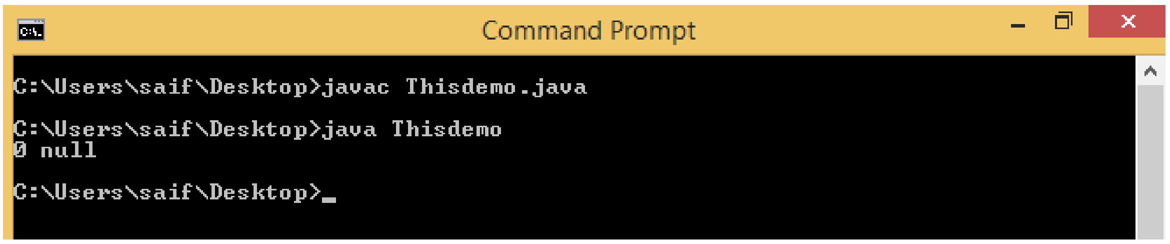
```
class Thisdemo
{
int roll;
String name;
public void Student(int roll , String name)
{
roll=roll;
name=name;
}
public void Show()
{
System.out.println(roll+" "+name);
}
public static void main(String[] args)
{
Thisdemo td=new Thisdemo();
td.Student(49,"Saiful");
```

td.Show();

}

}

Since we pass the value of roll no 49 and name “Saiful” from the object , so the expected output should be same roll no and name. But here the actual output is =>



```
C:\Users\saif\Desktop>javac Thisdemo.java

C:\Users\saif\Desktop>java Thisdemo
0 null

C:\Users\saif\Desktop>_
```

The output roll is 0 and name is null because because compiler is confused about the instant variable and the local variable of the class.So it print the value 0 and null.Since the variable of local and instant in the function student is same i.e. roll and name

To overcome this problem we used the this keyword in java.

In Java this keyword is used to

1. refer the current object
2. this keyword can be used to refer current class instance variable
3. If there is ambiguity between the instance variable and parameter, this keyword resolves the problem of ambiguity
4. this can be passed as an argument in the method call.
5. this can be passed as argument in the constructor call.
6. this keyword can also be used to return the current class instance.

Let’s see how this keyword helps us on the above ambiguity program. Now put the this keyword with instance variable.

So this is reference variable that refer to the current object of the program

class Thisdemo

{

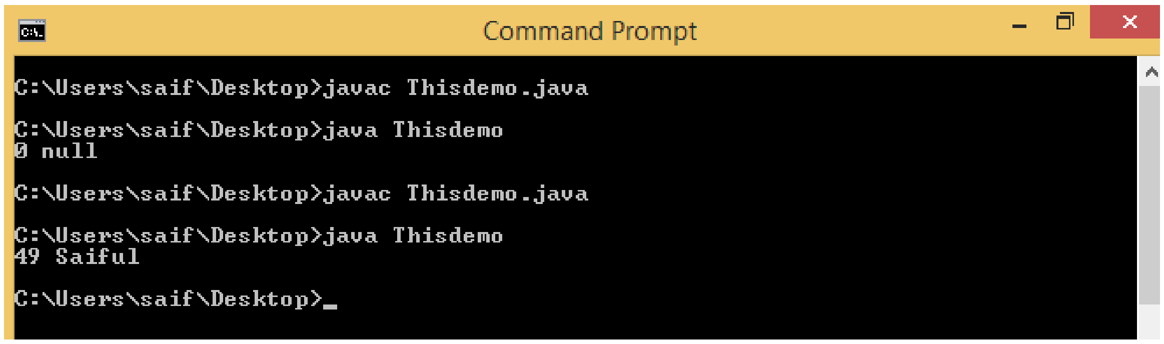
int roll;

String name;

public void Student(int roll , String name)

```
{
this.roll=roll;
this.name=name;
}
public void Show()
{
System.out.println(roll+" "+name);
}
public static void main(String[] args)
{
Thisdemo td=new Thisdemo();
td.Student(49,"Saiful");
td.Show();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Thisdemo.java

C:\Users\saif\Desktop>java Thisdemo
0 null

C:\Users\saif\Desktop>javac Thisdemo.java

C:\Users\saif\Desktop>java Thisdemo
49 Saiful

C:\Users\saif\Desktop>_
```

Use of the this() constructor:

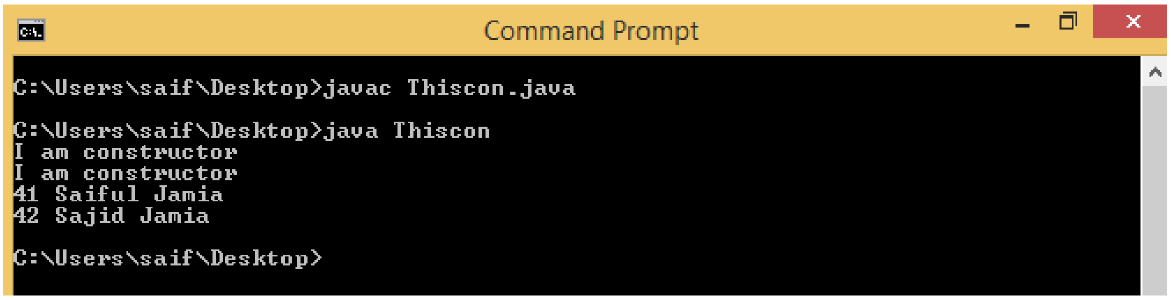
This can be used for the call the current class constructor

Program to use this() constructor

```
class Thiscon
```

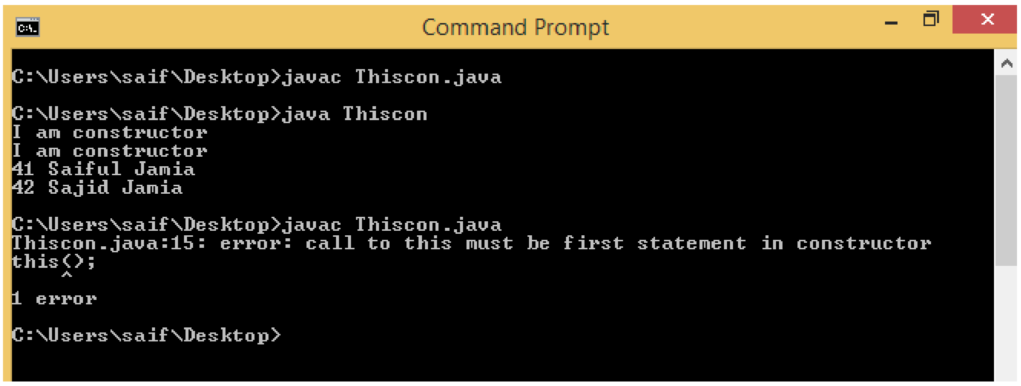
```
{
static String college="Jamia";
int roll;
String name;
Thiscon()
{
System.out.println("I am constructor");
}
Thiscon(int roll, String name)
{
this();
this.roll=roll;
this.name=name;
}
void display()
{
System.out.println(roll+" "+name+" "+college);
}
public static void main(String[] args)
{
Thiscon tc1=new Thiscon(41, "Saiful");
Thiscon tc2=new Thiscon(42, "Sajid");
tc1.display();
tc2.display();
}
}
```

In this program we have two constructor. One constructor is no value and one is parameterized constructor.To call the null constructor from null paramenter constructor we used the this() constructor. And the output will be as



```
C:\Users\saif\Desktop>javac Thiscon.java

C:\Users\saif\Desktop>java Thiscon
I am constructor
I am constructor
41 Saiful Jamia
42 Sajid Jamia

C:\Users\saif\Desktop>
```

Remember this() should be used at the first place inside the constructor otherwise you may have the following problem that **call to this must be first statement inside the constructor**

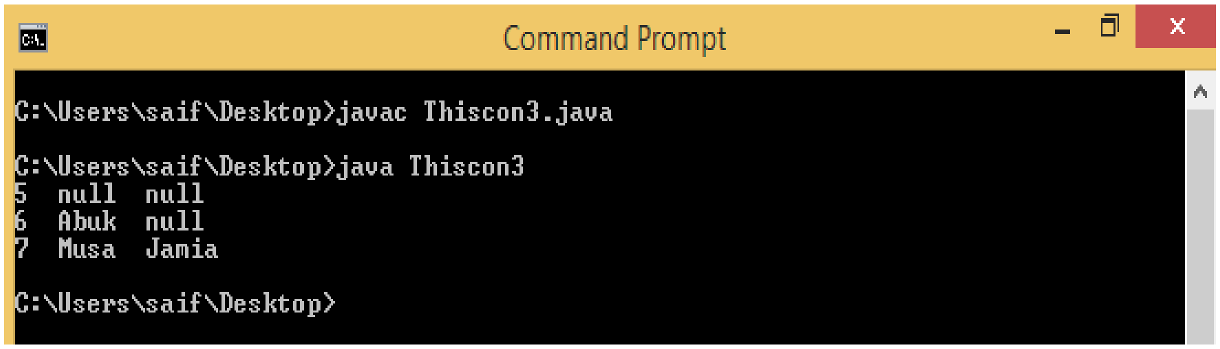


```
C:\Users\saif\Desktop>javac Thiscon.java

C:\Users\saif\Desktop>java Thiscon
I am constructor
I am constructor
41 Saiful Jamia
42 Sajid Jamia

C:\Users\saif\Desktop>javac Thiscon.java
Thiscon.java:15: error: call to this must be first statement in constructor
this();
    ^
1 error

C:\Users\saif\Desktop>
```

Program: How to use three constructors with this statement.

```
class Thiscon3
{
int roll;
String name;
String college;
Thiscon3(int roll)
{
this.roll=roll;
}
Thiscon3(int roll, String name)
{
this(roll);
this.name=name;
}
Thiscon3(int roll, String name, String college)
{
this(roll, name);
this.college=college;
}
public void display()
{
System.out.println(roll+"  "+name+ "  "+college);
}
public static void main(String[] args)
{
Thiscon3 tc31=new Thiscon3(5);
Thiscon3 tc32=new Thiscon3(6,"Abuk");
Thiscon3 tc33=new Thiscon3(7, "Musa","Jamia");
tc31.display();
tc32.display();
tc33.display();
}
}
```

Output;



```
C:\Users\saif\Desktop>javac Thiscon3.java

C:\Users\saif\Desktop>java Thiscon3
5  null  null
6  Abuk  null
7  Musa  Jamia

C:\Users\saif\Desktop>
```

Inheritance:

Inheritance simply mean the inherit or acquire the property of father or grandfather or forefather. With the help of this property child can have the parent of his father with additional property of him.

In Programming we used this in reusable code. If we have code in other class then instead of writing that code in new class we inherit the same code with the help of this property. From this the program becomes simple because there was no longer ambiguity of the same code. Generally in inheritance child class is called the sub class while the parent class is called the Super class.

We use the property of inheritance with the help of keyword “Extends”.

Property of inheritance:

1. Inheritance allows us to reuse the code in the program.
2. With the help of inheritance information is made manageable and in hierarchical order.
3. The class which inherit the property is known as the sub class while the other class is known as the Super class
4. “extends” is the keyword use to inherit the property of the Inheritance. With the help of this we reduce the code ambiguity and make the code more readable for the developer.
5. We also used inheritance in java for method overriding.

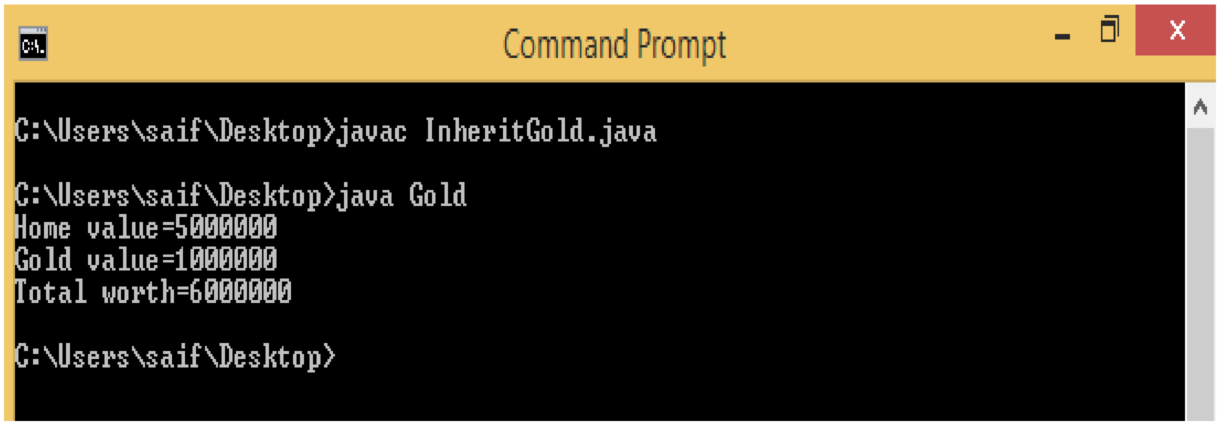
Program to demonstrate the Inheritance:

```
class Home

{

int Home=5000000;

}

class Gold extends Home

{

int Gold=1000000;
void display()
{
System.out.println("Home value="+Home);
System.out.println("Gold value="+Gold);
int s=Gold+Home;
System.out.println("Total worth="+s);
}
public static void main(String[] args)
{
Gold g=new Gold();
g.display();
}
}
```

Output:

The base class have the property of gold of rs 100000 while he also inherit the property of his father home which value is Rs. 500000.

So after inherit the super class money bease class have the total property of Rs. 6000000.



```
C:\Users\saif\Desktop>javac InheritGold.java

C:\Users\saif\Desktop>java Gold
Home value=5000000
Gold value=1000000
Total worth=6000000

C:\Users\saif\Desktop>
```

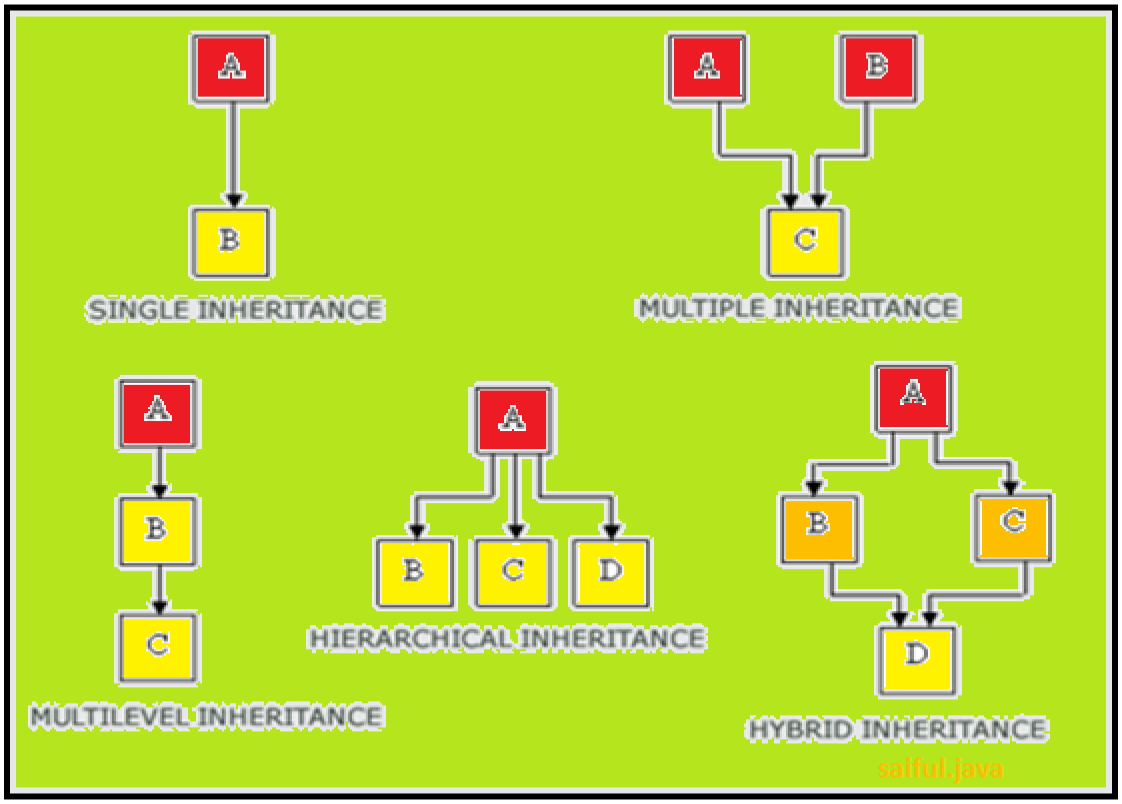
Types Of Inheritance:

There are three types of inheritance in java

1. Simple inheritance

2. Multilevel inheritance
3. Hierachical Inheritance
4. Hybrid Inheritance

Multiple inheritance is not supported in java because of its theory of simplicity but we can achieve it by the help of interface.



Single level Inheritance:

It is very simple type of inheritance in which a sub class acquire the property of the super class.

Syntax:

Class A

{

…………..

}

Class B extends Class A

{

…..

}

See the above example

Multilevel inheritances:

In multiple inheritances a sub class acquire the property of super class which is already a sub class.

Syntax:

Class A

{

…………..

}

Class B extends A

{

…………..

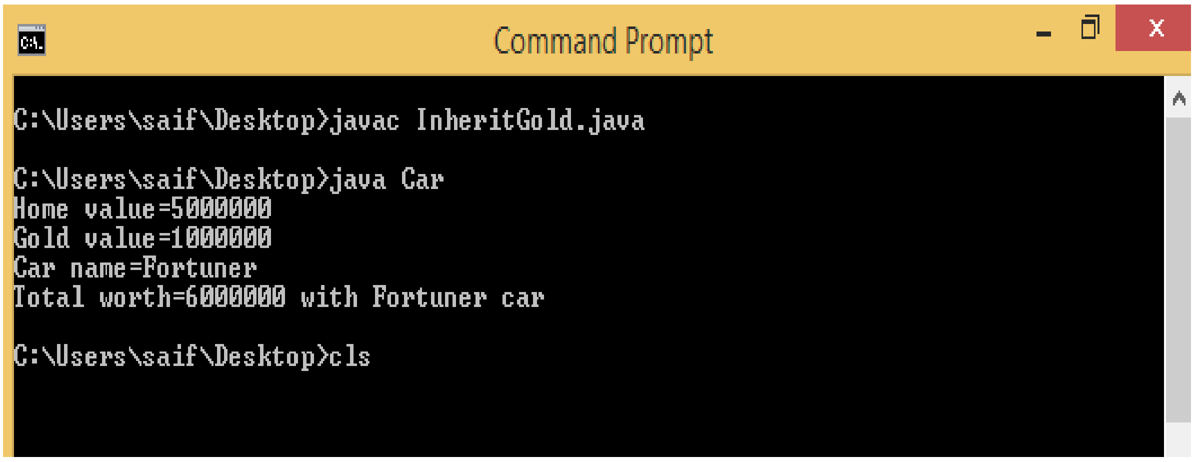
}

Class C extends B // class B is already a sub class but class C inherit the pproperty of class B as well as class A by making the class B a super class for class C.

Program for multilevel inheritance:

```
class Home

{

int Home=5000000;

}

class Gold extends Home

{

int Gold=1000000;

}

class Car extends Gold

{

String car="Fortuner";
void display()
{
System.out.println("Home value="+Home);
System.out.println("Gold value="+Gold);
System.out.println("Car name="+car);
int s=Gold+Home;
System.out.println("Total worth="+s+" with "+car+" car");
}
public static void main(String[] args)
{
Car c=new Car();
c.display();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac InheritGold.java

C:\Users\saif\Desktop>java Car
Home value=5000000
Gold value=1000000
Car name=Fortuner
Total worth=6000000 with Fortuner car

C:\Users\saif\Desktop>cls
```

Hierarchical Inheritance

In hierarchical inheritance one class can have two more than two super class.

Syntax:

```
Class A
{
……………….
}
Class B extends A
{
…………….
}
Class C extends A
{
………………..
}
Class D extends A
{
………………..
}
```

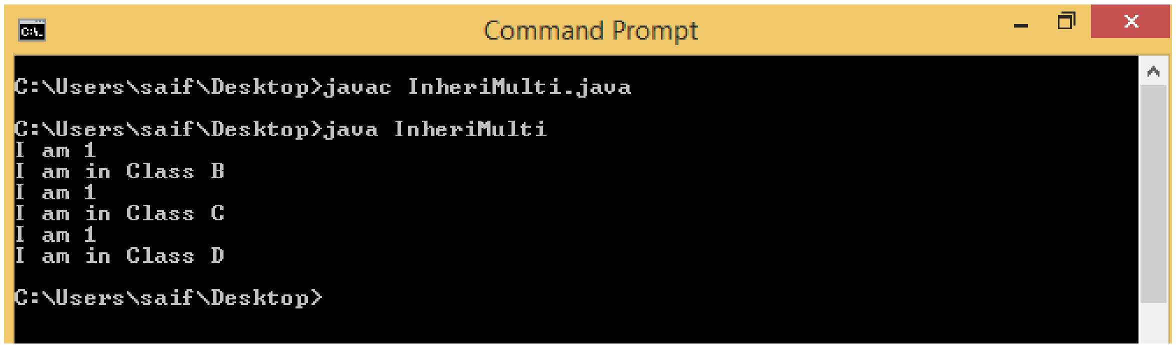
So class A extends by more than two classes. So hierarchical inheritance is here.

Program on hierarchical inheritance:

```
class A
{
int a=1;
}
class B extends A
{
void displayB()
{
System.out.println("I am "+a);
System.out.println("I am in Class B");
}
}
class C extends A
{
void displayC()
{
System.out.println("I am "+a);
System.out.println("I am in Class C");
}
}
class D extends A
{
void displayD()
{
System.out.println("I am "+a);
System.out.println("I am in Class D");
}
}
class InheriMulti
{
public static void main(String[] args)
{
B d1=new B();
C d2=new C();
D d3=new D();
d1.displayB();
d2.displayC();
```

```
d3.displayD();
}
}
```

Output:


Command Prompt

```
C:\Users\saif\Desktop>javac InheriMulti.java

C:\Users\saif\Desktop>java InheriMulti
I am 1
I am in Class B
I am 1
I am in Class C
I am 1
I am in Class D

C:\Users\saif\Desktop>
```

So class A is extends by the more than two classes with the same result.

Overriding:

Declaring a method in sub class which is already present in the super class is known as the method overriding.

If we want to change the property of our parent class or want to replace it from the base class then we use the Overriding method.

Properties of the overriding method:

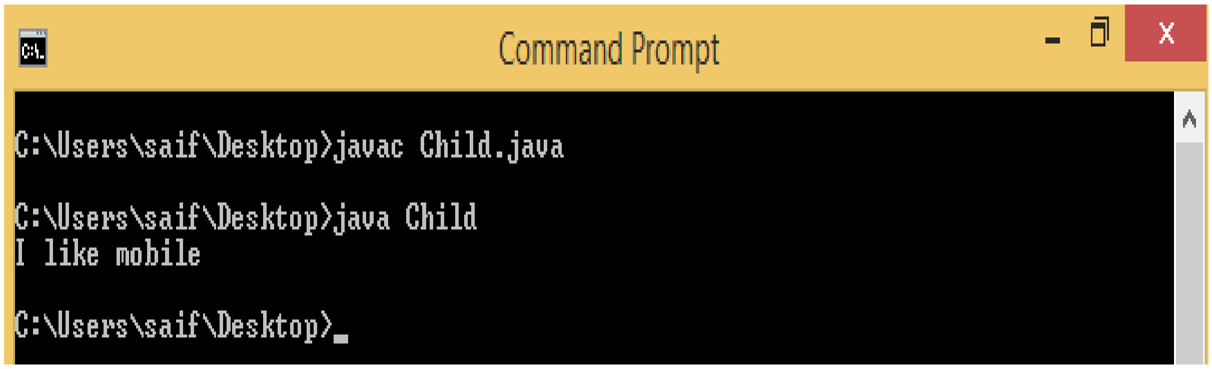
1. It applies only to the inherited method.
2. Object type determines which overridden method will be used only at run time.
3. Abstract method must be overridden.
4. Final method cannot be overridden
5. Static method cannot be overridden
6. Constructor cannot be overridden
7. Overriding method must-have the same number of argument and data types
8. Overriding method must have the sane return data type
9. Overriding method cannot have the more restrictive access but can have less restrictive access
10. Overriding method must not through new or broader checked exception but may throw fewer or narrower checked exception or any unchecked exception
11. Overriding is the run time polymorphism in java.
12. In overriding there is a IS-A relationship between the classes.
13. Sonce main method is static, so we cannot override the main method.

Method overriding is used to provide specific implementation in a method that is already provided by its super class.

Program on overriding

```
class Father
{
void display()
{
System.out.println("Father dont like mobile");
}
}
class Child extends Father
{
void display()
{
System.out.println("I like mobile ");
}
public static void main(String[] args)
{
Child c=new Child();
c.display();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Child.java

C:\Users\saif\Desktop>java Child
I like mobile

C:\Users\saif\Desktop>_
```

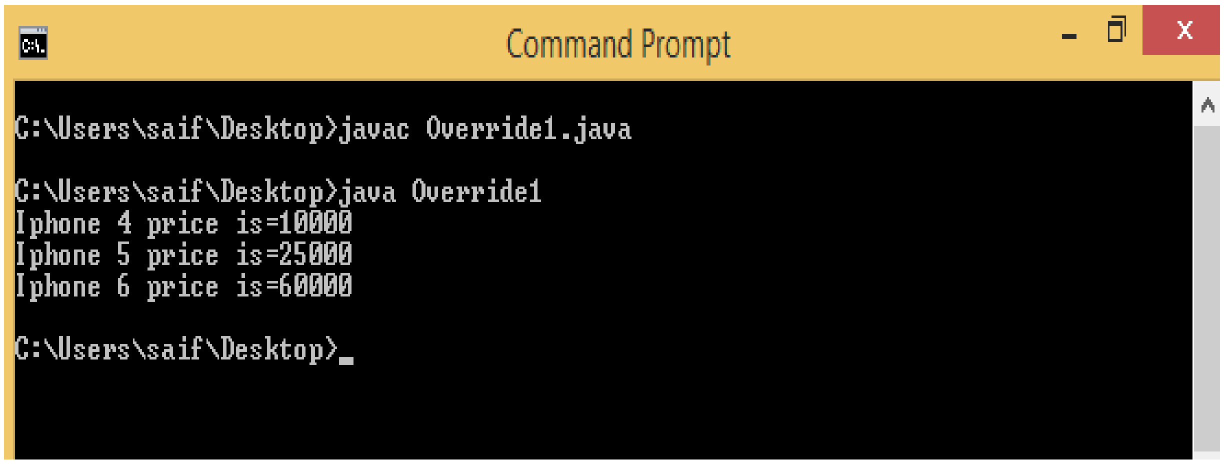
Benefit of method overriding:

The main advantage of method overriding is that the sub class can give its own feature to the inherited method from the super class.

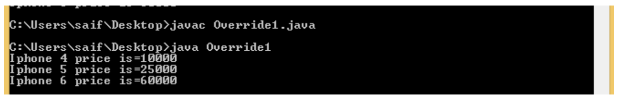
Program of override:

```
class Iphone
{
int rate()
{
return 0;
}
}
class Iphone4 extends Iphone
{
int rate()
{
return 10000;
}
}
class Iphone5 extends Iphone
{
int rate()
{
return 25000;
}
}
class Iphone6 extends Iphone
{
int rate()
{
return 60000;
}
}
class Override1
{
public static void main(String[] args)
{
Iphone4 i4=new Iphone4();
Iphone5 i5=new Iphone5();
Iphone6 i6=new Iphone6();
System.out.println("Iphone 4 price is="+i4.rate());
System.out.println("Iphone 5 price is="+i5.rate());
System.out.println("Iphone 6 price is="+i6.rate());
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Override1.java

C:\Users\saif\Desktop>java Override1
Iphone 4 price is=10000
Iphone 5 price is=25000
Iphone 6 price is=60000

C:\Users\saif\Desktop>_
```



```
C:\Users\saif\Desktop>javac Override1.java

C:\Users\saif\Desktop>java Override1
Iphone 4 price is=10000
Iphone 5 price is=25000
Iphone 6 price is=60000
```

Super Keyword in Java:

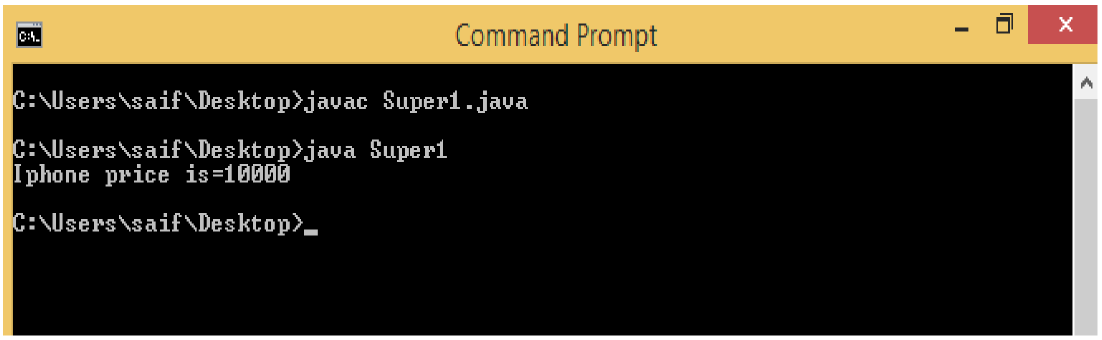
1. As implies by its name super means call the super or parent.
2. Super keyword in Java is used to distinguish between parent class instance variable and the child class instance variable
3. It is used to call the any function/ constructor of its class
4. If there is the same method or instance variable then super class is used to distinguish between parent class and base class.
5. Superkeyword is used to refer immediate parent class instance variable.
6. Super is used to invoke immediate parent class constructor.
7. Super is used to invoke immediate parent class method.

Program: Program without super class

```
class Apple
{
int r=15000;
}
class Iphone extends Apple
{
int r=10000;
public void Price()
{
System.out.println("Iphone price is="+r);
}
}
class Super1
{
public static void main(String[] args)
{
Iphone i=new Iphone();
i.Price();
}
}
```

Output:

Here the price of Iphone is Rs. 10,000 which is the child class Iphone price



```
C:\Users\saif\Desktop>javac Super1.java

C:\Users\saif\Desktop>java Super1
Iphone price is=10000

C:\Users\saif\Desktop>_
```

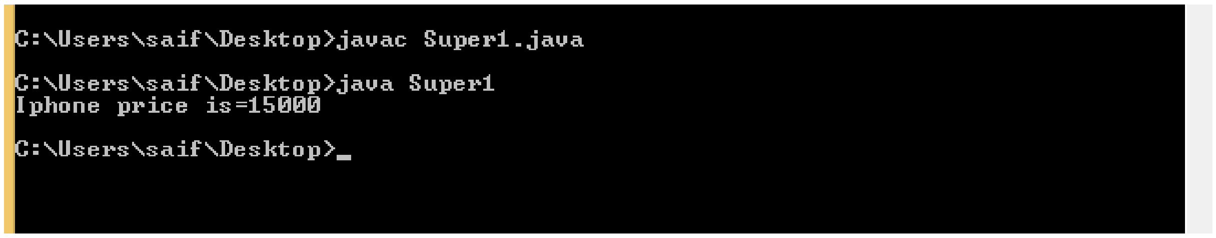
However if we want to print the price of Apple then we there would ambiguity because of the similar integer variable in both the parent and child class. So we use the the “this” keyword to reference the variable of the parent class.

Program to demonstrate the use of thiskeyword

```
class Apple
{
int r=15000;
}
```

```
class Iphone extends Apple
{
int r=10000;
public void Price()
{
System.out.println("Iphone price is="+super.r);
}
}
class Super1
{
public static void main(String[] args)
{
Iphone i=new Iphone();
i.Price();
}
}
```

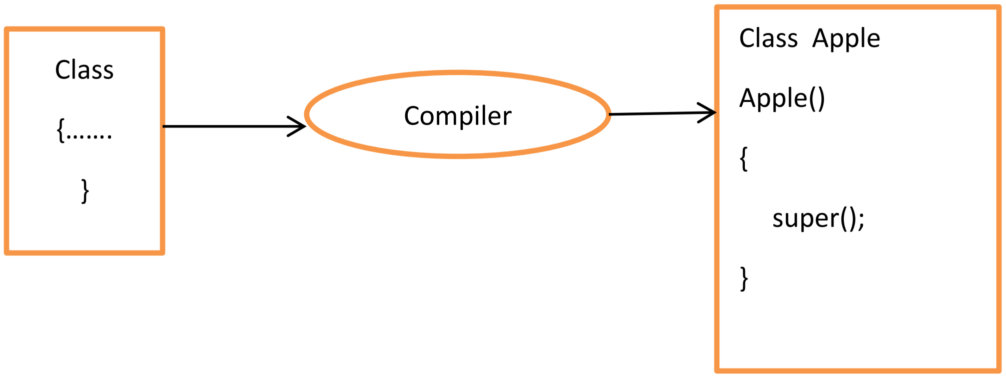
Output:



```
C:\Users\saif\Desktop>javac Super1.java

C:\Users\saif\Desktop>java Super1
Iphone price is=15000

C:\Users\saif\Desktop>_
```

So this helps us to access the value of the parent class.

Compiler always provide the super() as the first class constructor.

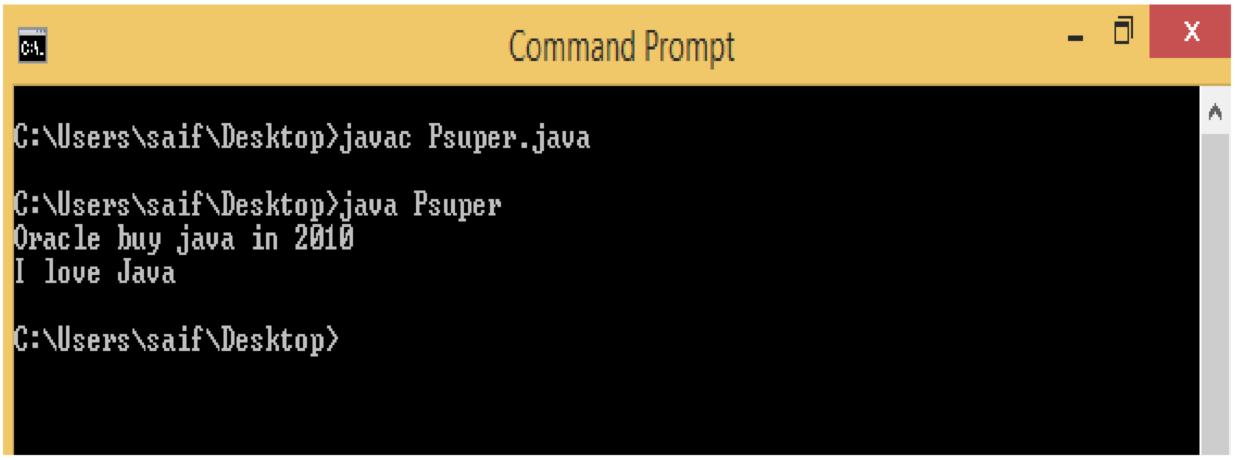


Super is also used to call the parent class method.

Program to demonstrate super is used to call the parent class ethod.

```
class Oracle
{
int i=2010;
void display()
{
System.out.println("Oracle buy java in "+i);
}
}
class Java extends Oracle
{
void display()
{
super.display();
System.out.println("I love Java");
}
}
class Psuper
{
public static void main(String[] args)
{
Java j=new Java();
j.display();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Psuper.java

C:\Users\saif\Desktop>java Psuper
Oracle buy java in 2010
I love Java

C:\Users\saif\Desktop>
```

Final Keyword:

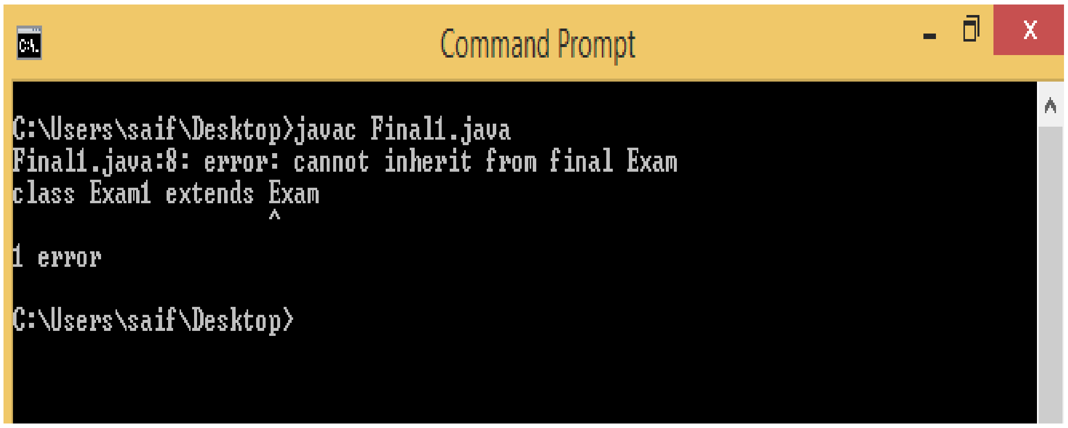
Final keyword in Java is used to make the entity non modifiable.It is used in the following context. It is used to restrict changing of class variable, method and the instance as well as local variable.

1. If we declare the class as final then we can not inherit the property in sub class. Program to show the error how can a final class can not be inherit from the base class

```
final class Exam
{
void display()
{
System.out.println("Exam date is 12/5/2016");
}
}
class Exam1 extends Exam
{
void display()
{
super.display();
System.out.println("Exam1 time is 9.30 AM");
}
}
class Final1
{
public static void main(String[] args)
{
```

```
Exam1 e=new Exam1();
e.display();
}
}
```

Output:

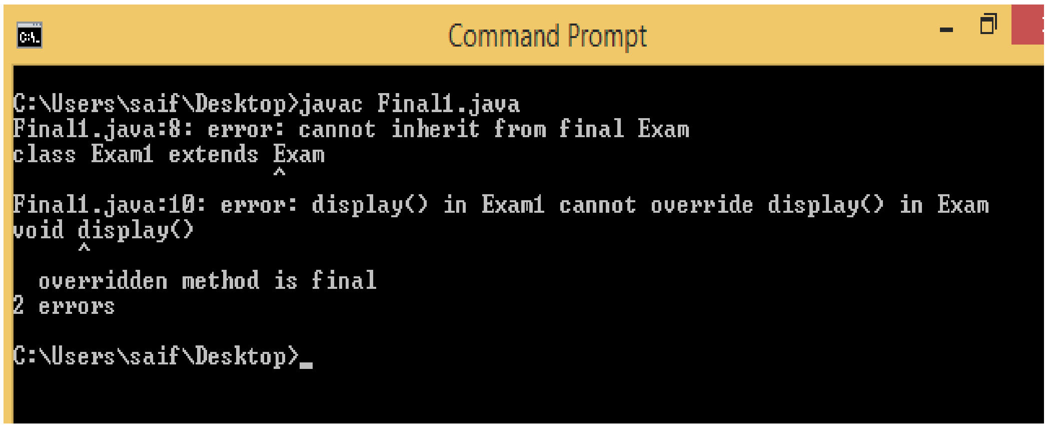


```
C:\Users\saif\Desktop>javac Final1.java
Final1.java:8: error: cannot inherit from final Exam
class Exam1 extends Exam
                    ^
1 error

C:\Users\saif\Desktop>
```

2. Method also declares as the final method. Method declare as a final method cannot be override by the property of inheritance.
   Program to demonstrate:

```
final class Exam
{
final void display()
{
System.out.println("Exam date is 12/5/2016");
}
}
class Exam1 extends Exam
{
void display()
{
System.out.println("Exam1 time is 9.30 AM");
}
}
class Final1
{
public static void main(String[] args)
{
Exam1 e=new Exam1();
e.display();
}
}
```

Output:

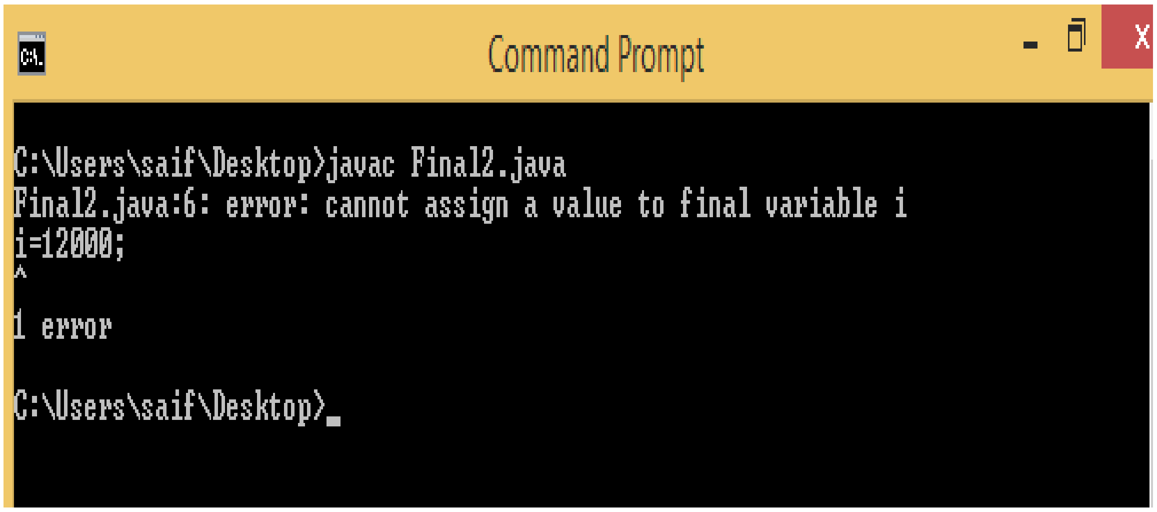

```
C:\Users\saif\Desktop>javac Final1.java
Final1.java:8: error: cannot inherit from final Exam
class Exam1 extends Exam
                    ^
Final1.java:10: error: display() in Exam1 cannot override display() in Exam
void display()
     ^
  overridden method is final
2 errors

C:\Users\saif\Desktop>_
```

So, the final cannot be inherited by the sub class.

3. Final keyword is used to make the variable constant whether the static or reference variable.

```
class Final2
{
final int  i=10000;
public void show()
{
i=12000;
System.out.println("Iphone 4 price is="+i);
}
public static void main(String[] args)
{
Final2 f=new Final2();
f.show();
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Final2.java
Final2.java:6: error: cannot assign a value to final variable i
i=12000;
^
1 error

C:\Users\saif\Desktop>_
```

Uninitialized or Blank final variable:

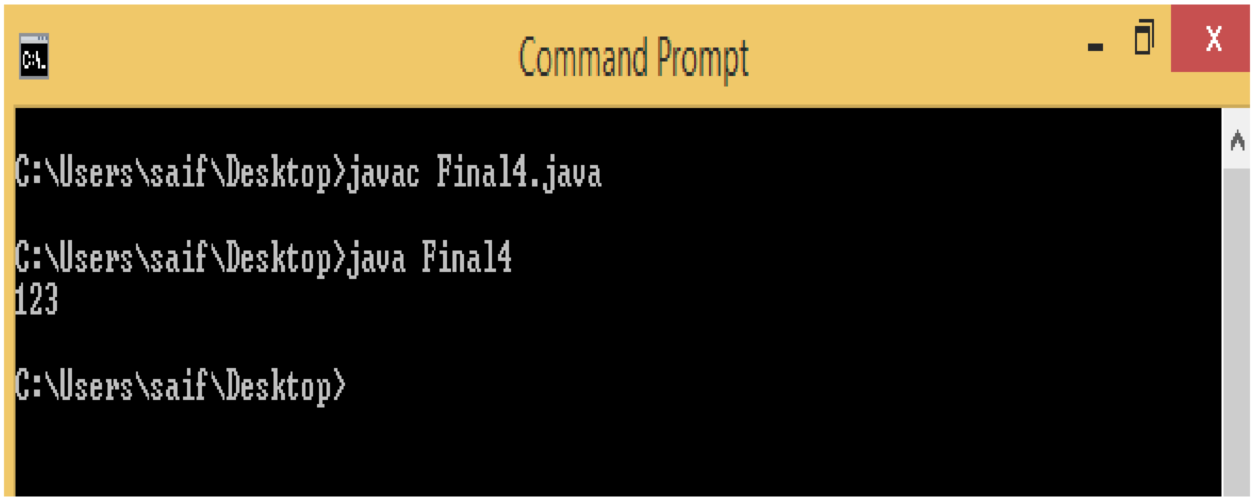
A final variable that is not initialized at that time of declaration is known as the black final variable.

We can initialize the bank final variable after the declaration with the help of constructor only.

Program to initialize the value into blank final variable by the help of constructor

```
class Final4
{
 final int pan;
      Final4()
        {
        pan=123;
        System.out.println(pan);
        }
public static void main(String[] args)
        {
        new Final4();
        }
}
```

Output:

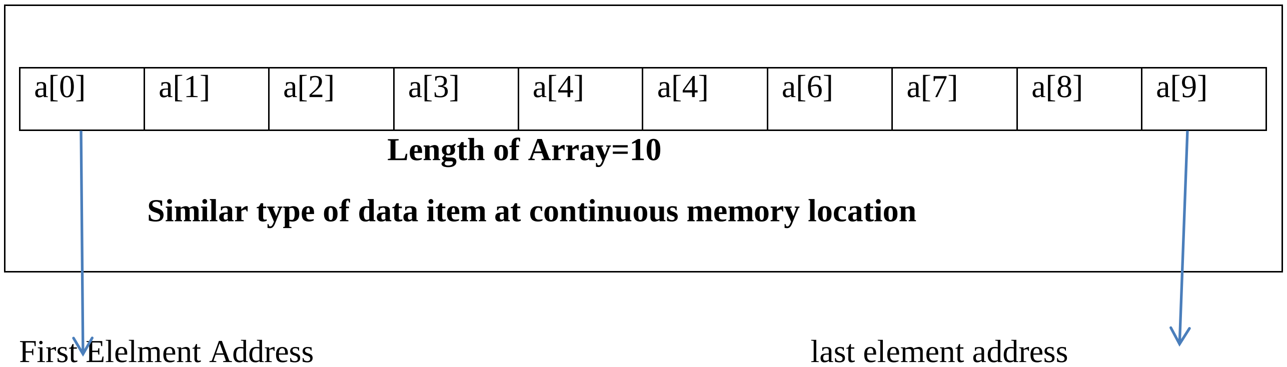


```
C:\Users\saif\Desktop>javac Final4.java

C:\Users\saif\Desktop>java Final4
123

C:\Users\saif\Desktop>
```

Note: We cannot make the constructor final because the constructor is never inherited. Constructor cannot be overridden.

Array:

Array is a collection of similar type of data item at contiguous memory location. Array is based on the index. It is used to help in sorting; merging of data. Array Is based on the index. It just contains the address of the element. The first element of the array store at array [0] location while the last element of the array store at array[n-1] location where the n is the length of the array.



First Elelment Address last element address

Program on Array: How array store the data

import java.util.*;

class Array1

{

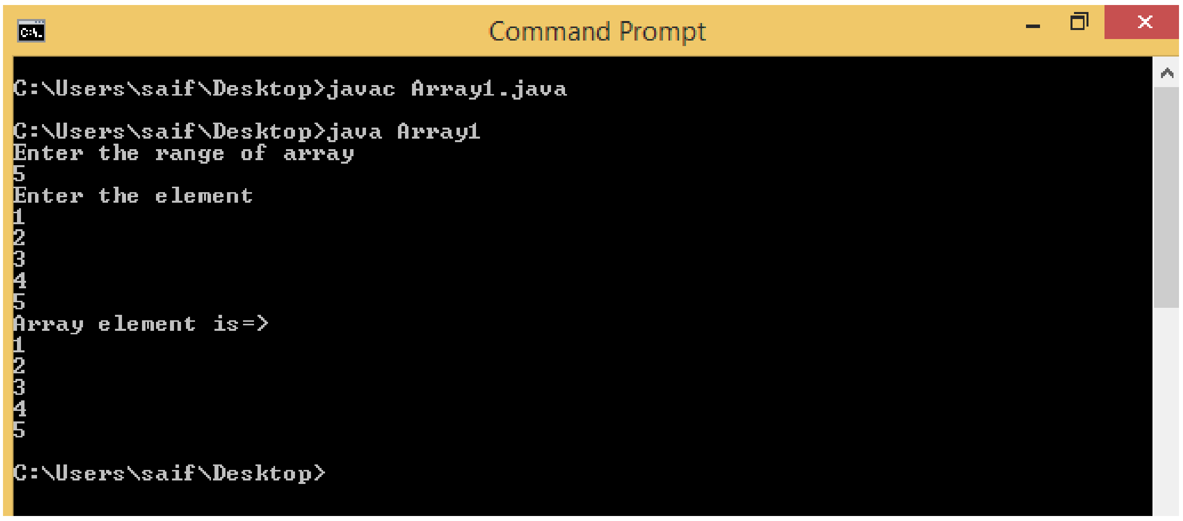
public static void main(String[] args)

{

Scanner in=new Scanner(System.in);

```
int i=0;
int a[]=new int[100];
System.out.println("Enter the range of array");
int m=in.nextInt();
System.out.println("Enter the element");
for(i=0;i<m;i++)
a[i]=in.nextInt();
System.out.println("Array element is=>");
for(i=0;i<m;i++)
{
System.out.println(+a[i]);
}
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Array1.java

C:\Users\saif\Desktop>java Array1
Enter the range of array
5
Enter the element
1
2
3
4
5
Array element is=>
1
2
3
4
5

C:\Users\saif\Desktop>
```

In java arrary usually use the integer for creating index.The maximum length of the java array is 2^31-1 =2147483647

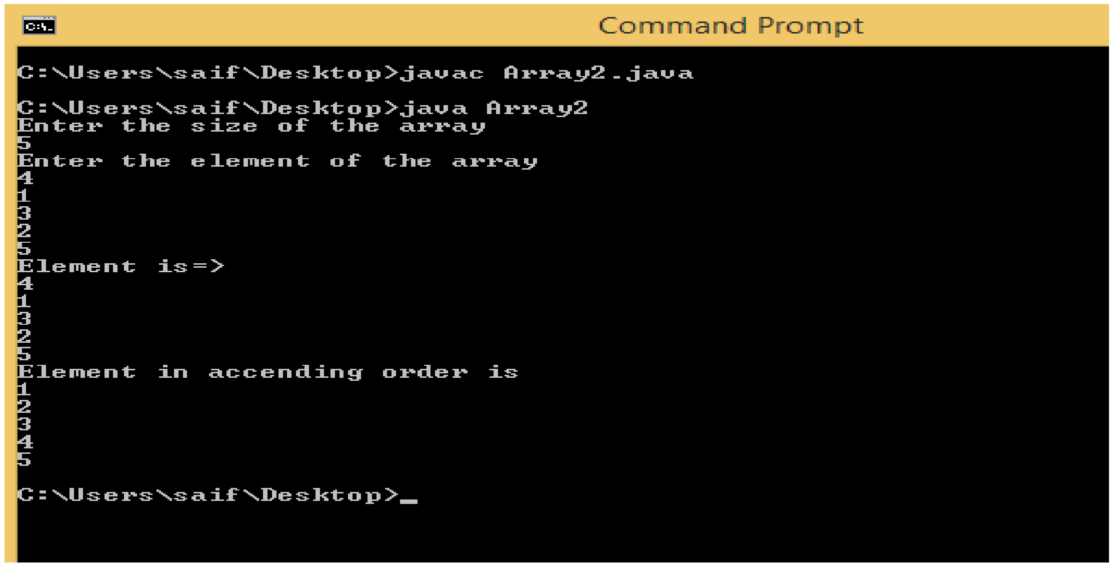
Program on sorting of array:

Sorting of array in ascending order:

```
import java.util.*;
class Array2
        {
          public static void main(String[] args)
               {
                int i, m;
                int a[]=new int[100];
                Scanner in=new Scanner(System.in);
                System.out.println("Enter the size of the array");
                m=in.nextInt();
                System.out.println("Enter the element of the array");
               for(i=0;i<m;i++)
                 {
                  a[i]=in.nextInt();
                 }
                  System.out.println("Element is=>");
                  for(i=0;i<m;i++)
                      {
                      System.out.println(a[i]);
                      }
                 for(i=0;i<m-1;i++)
                     {
                      for(int j=i+1;j<m;j++)
                          {
                          if(a[i]>a[j])
                           {
                           int temp=a[i];
                           a[i]=a[j];
```

```
                        a[j]=temp;
                        }
                       }
                   }
           System.out.println("Element in accending order is");
           for(i=0;i<m;i++)
              {
               System.out.println(a[i]);
              }
            }
         }
```

Output:



```
C:\Users\saif\Desktop>javac Array2.java

C:\Users\saif\Desktop>java Array2
Enter the size of the array
5
Enter the element of the array
4
1
3
2
5
Element is=>
4
1
3
2
5
Element in accending order is
1
2
3
4
5

C:\Users\saif\Desktop>_
```

Advantage of array:

1. It is used store multiple data items of same type by using only single name.
2. Any element can be randomly access using the indexes.
3. It can be used to implement other data structures like linked lists, stacks, queues, trees, graphs etc.
4. It helps us in sorting and merging of the data.
5. 2D arrays are used to represent matrices.

Disadvantages of the array:

1. We cannot modify the size of the array. The memory which is allocated to the array cannot be reduce or increased.
2. If we allocate the memory then requirement the memory will be waste and if we take less size then it also create the problem.
3. Array store the data at consecutive memory location. So it is time consuming to insert and merge the data.

Array is of two type:

1. Single dimensional array like linked list, stack and queue
2. Multidimensional Array like Matrix

Syntax to determine the Array:

Int varname []=new int[var size];

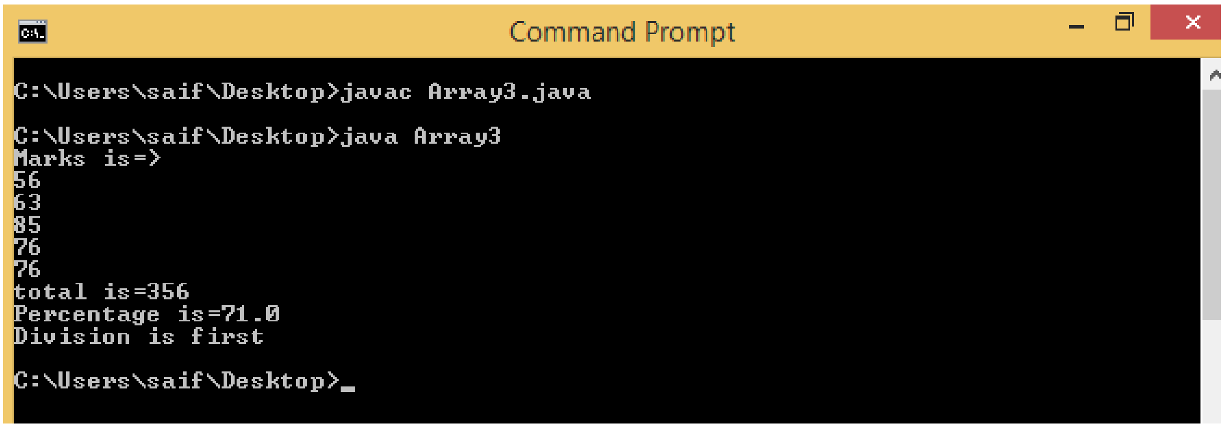
Int [] varname=new int[var size];

Int a[]={1,2,4,5,7};

Program on Declaration of Array

```
class Array3

{

public static void main(String[] args)

{

int a[]={56,63,85,76,76};

int total=0;

System.out.println("Marks is=>");

for(int i=0;i<a.length;i++)

{

System.out.println(+a[i]);

}

for(int i=0;i<a.length;i++)

{

total=total+a[i];

}
System.out.println("total is="+total);
float f=total/a.length;
System.out.println("Percentage is="+f);
if(f>60)
    {
    System.out.println("Division is first");
    }
    else if(f<60 && f>45)
       {
       System.out.println("Division is second");
       }
    else
       {
       System.out.println("Division is third");
       }
}
}
```

Output :



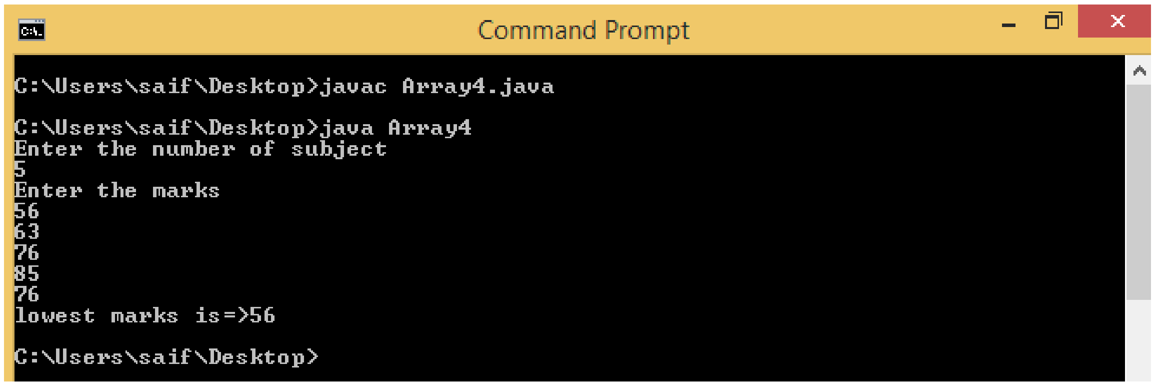
```
C:\Users\saif\Desktop>javac Array3.java

C:\Users\saif\Desktop>java Array3
Marks is=>
56
63
85
76
76
total is=356
Percentage is=71.0
Division is first

C:\Users\saif\Desktop>_
```

Program to print the lowest number in the array:

```
import java.util.*;
```

```
class Array4
{
public static void main(String[] args)
{
int i;
int a[]=new int[10];
Scanner in=new Scanner(System.in);
System.out.println("Enter the number of subject");
int m=in.nextInt();
System.out.println("Enter the marks");
for(i=0;i<m;i++)
{
a[i]=in.nextInt();
}
int min=a[0];
for(i=1;i<m;i++)
{
if(min>a[i])
{
int temp=a[i];
a[i]=min;
min=temp;
}
}
System.out.println("lowest marks is=>"+min);
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Array4.java

C:\Users\saif\Desktop>java Array4
Enter the number of subject
5
Enter the marks
56
63
76
85
76
lowest marks is=>56

C:\Users\saif\Desktop>
```

Multidimensional Array:

A multidimensional array is one which store the data in the form of row and column i.e. in the form of matrix.

Syntax to multidimensional array

Int a[][]=new a[m][n];

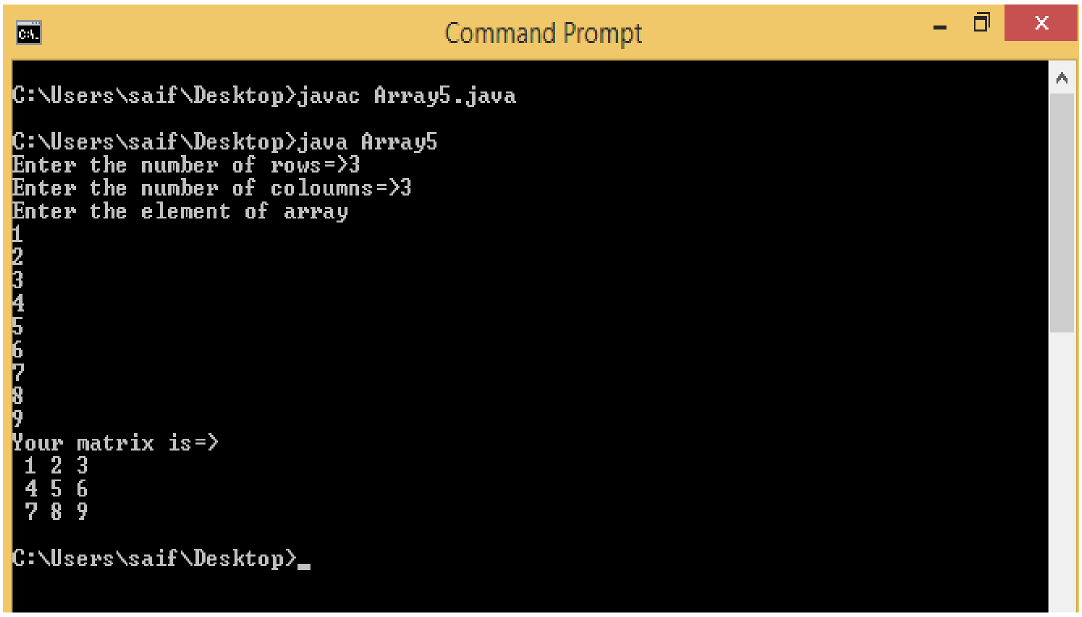
int arr[][]={{1,2,3},{2,4,5},{4,4,5}};

Program to print the matrix:

```
import java.util.*;

class Array5

{

 public static void main(String[] args)

   {

   Scanner in=new Scanner(System.in);

   int i,j,m,n;

   int a[][]=new int[10][10];

   System.out.print("Enter the number of rows=>");

   m=in.nextInt();

   System.out.print("Enter the number of coloumns=>");

   n=in.nextInt();

   System.out.println("Enter the element of array");

   for(i=0;i<m;i++)
```

```
      {
      for(j=0;j<n;j++)
          {
           a[i][j]=in.nextInt();
          }
      }
   System.out.println("Your matrix is=>");
      for(i=0;i<m;i++)
      {
       for(j=0;j<n;j++)
          {
         System.out.print(" ");
         System.out.print(a[i][j]);
          }
       System.out.println("");
      }
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Array5.java

C:\Users\saif\Desktop>java Array5
Enter the number of rows=>3
Enter the number of coloumns=>3
Enter the element of array
1
2
3
4
5
6
7
8
9
Your matrix is=>
 1 2 3
 4 5 6
 7 8 9

C:\Users\saif\Desktop>_
```

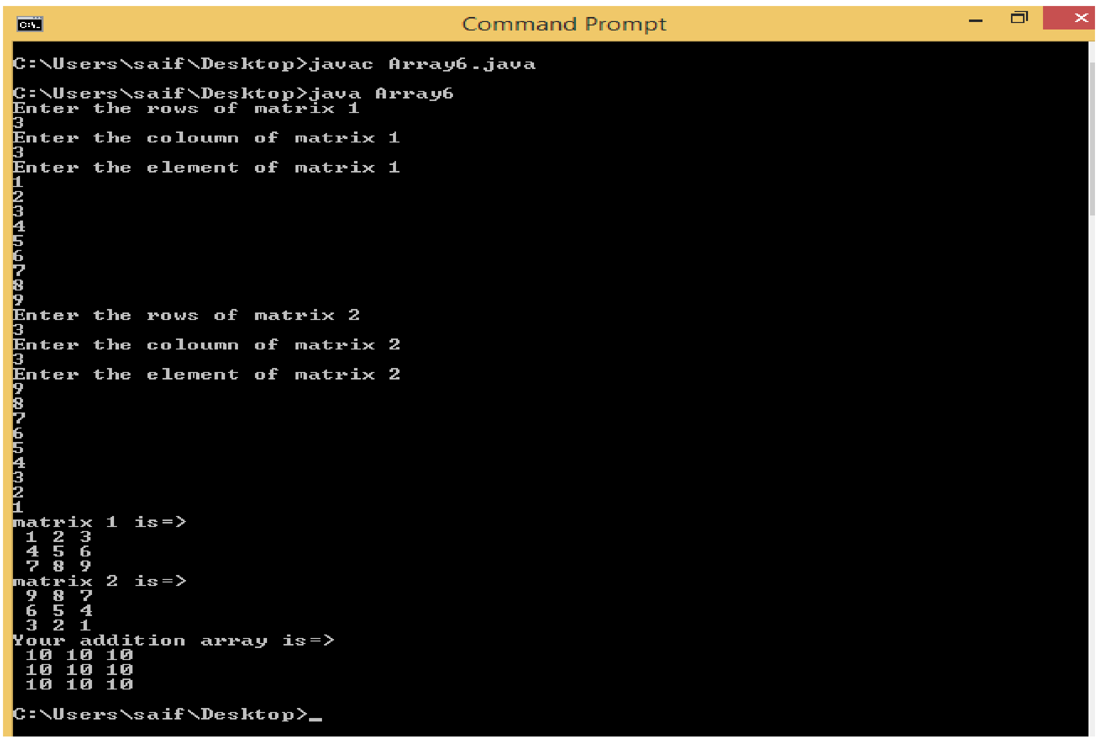
Matrix Addition: Program on matrix addition of two number

```
import java.util.*;
class Array6
{
public static void main(String[] args)
{
int i,j,m1,n1,m2,n2,m,n;
int a[][]=new int[10][10];
int b[][]=new int[10][10];
int c[][]=new int[10][10];
Scanner in=new Scanner(System.in);
System.out.println("Enter the rows of matrix 1");
m1=in.nextInt();
System.out.println("Enter the coloumn of matrix 1");
n1=in.nextInt();
System.out.println("Enter the element of matrix 1");
for(i=0;i<m1;i++)
{
```

```
for(j=0;j<n1;j++)
{
a[i][j]=in.nextInt();
}
}
System.out.println("Enter the rows of matrix 2");
m2=in.nextInt();
System.out.println("Enter the coloumn of matrix 2");
n2=in.nextInt();
System.out.println("Enter the element of matrix 2");
for(i=0;i<m2;i++)
{
for(j=0;j<n2;j++)
{
b[i][j]=in.nextInt();
}
}
System.out.println("matrix 1 is=>");
for(i=0;i<m1;i++)
{
for(j=0;j<n1;j++)
{
System.out.print(" ");
System.out.print(+a[i][j]);
}
System.out.println("");
}
System.out.println("matrix 2 is=>");
```

```
for(i=0;i<m2;i++)
{
for(j=0;j<n2;j++)
{
System.out.print(" ");
System.out.print(+b[i][j]);
}
System.out.println("");
}
if(m1==m2 && n1==n2)
{
m=m1;
n=n1;
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
c[i][j]=a[i][j]+b[i][j];
}
}
System.out.println("Your addition array is=>");
for(i=0;i<m1;i++)
{
for(j=0;j<n1;j++)
{
System.out.print(" ");
System.out.print(+c[i][j]);
}
System.out.println("");
}
}
else
{
System.out.println("two Array can be add for same number of rows and coloumn");
}
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Array6.java

C:\Users\saif\Desktop>java Array6
Enter the rows of matrix 1
3
Enter the coloumn of matrix 1
3
Enter the element of matrix 1
1
2
3
4
5
6
7
8
9
Enter the rows of matrix 2
3
Enter the coloumn of matrix 2
3
Enter the element of matrix 2
9
8
7
6
5
4
3
2
1
matrix 1 is=>
 1 2 3
 4 5 6
 7 8 9
matrix 2 is=>
 9 8 7
 6 5 4
 3 2 1
Your addition array is=>
 10 10 10
 10 10 10
 10 10 10

C:\Users\saif\Desktop>_
```

Wrapper class:

Java is 99% object oriented language. The lack of 1% is due to the primitive data types such as character, int , float , double , byte , long, boolean and short. Just because of these primitive data type Java is not called 100 % object oriented language. But if you want to work with Java as

100% object oriented language then Java allow us to use the classes for every data type from char to short. These classes are called the wrapper classes.

The 8 primitive types and its wrapper classes are,

byte.	**- Byte**
int	**- Integer**
short	**- Short**
long	**- Long**
float	**- Float**
double	**- Double**
char	**- Character**
boolean	**- Boolean**

These all wrapper class is available in the java.lang package.

Wrapper class is used to represent the primitive values when the object is required.When the primitive data types is converted into object then we have to dealt with only object in our program and making it 100% object oriented.

Use of wrapper class:

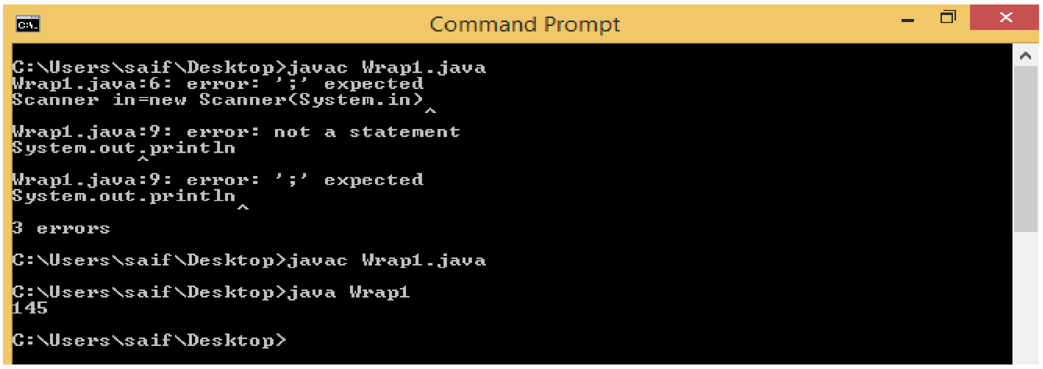
If you want to create a LinkedList of Integers, for example, using LinkedList<int> will give you a compile error. Instead, you have to use the wrapper class Integer (e.g. LinkedList<Integer>)

Note: Wrapper is not the class but it is a group of classes in Java.lang package for primitive data type.

But after the release of the J2SE 5.0, Auto boxing and unboxing feature convert primitive into object and object into primitive data type automatically.

Method of Wrapper class:

1. **valueOf():** It is a static method which return the object reference of the relative wrapper class.
   Program for example:
   ```
   import java.util.*;
   class Wrap1
   {
   public static void main(String[] args)
   {
   Integer b=Integer.valueOf("145");
   System.out.println(b);
   }
   }
   ```
   Output:

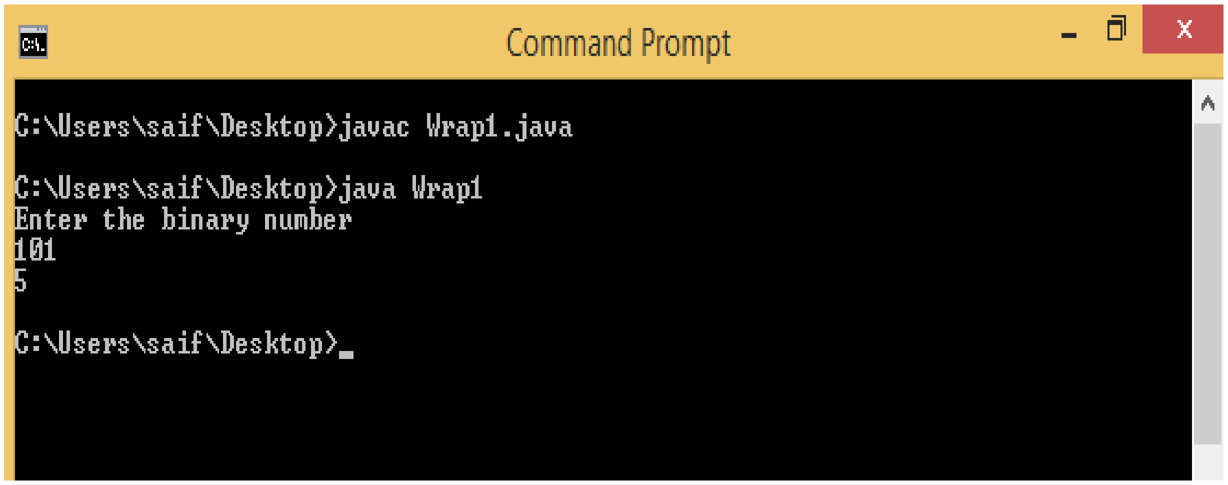


```
C:\Users\saif\Desktop>javac Wrap1.java
Wrap1.java:6: error: ';' expected
Scanner in=new Scanner(System.in)
                                 ^
Wrap1.java:9: error: not a statement
System.out.println
          ^
Wrap1.java:9: error: ';' expected
System.out.println
                  ^
3 errors

C:\Users\saif\Desktop>javac Wrap1.java

C:\Users\saif\Desktop>java Wrap1
145

C:\Users\saif\Desktop>
```

Program to conver binary number into decimal with the help of wrapper class.

```
import java.util.*;
class Wrap1
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the binary number");
String b=in.next();
Integer a=Integer.valueOf(b,2);
System.out.println(a);
}
}
```

Output:

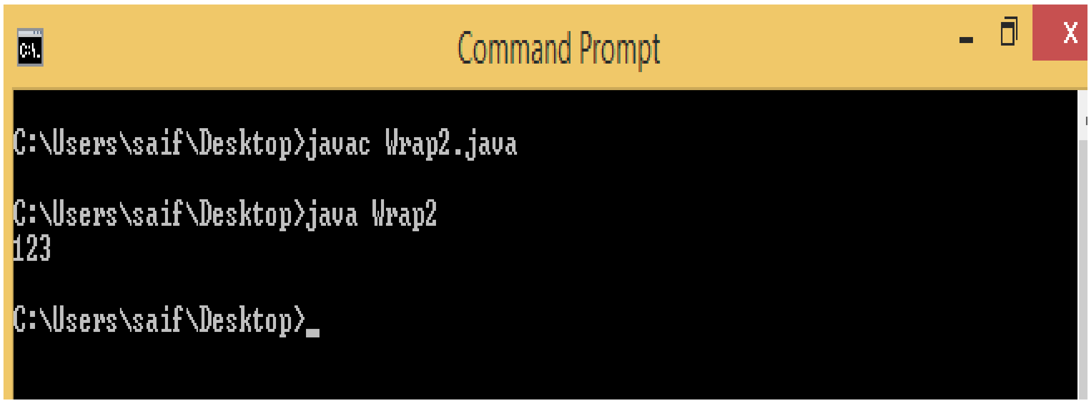


```
C:\Users\saif\Desktop>javac Wrap1.java

C:\Users\saif\Desktop>java Wrap1
Enter the binary number
101
5

C:\Users\saif\Desktop>_
```

2. **parseXxx(); :**Second important function of wrapper class is parseXxx();. It is a static method where Xxx is a primitive type value such as parseInt();; parseFloat(); etc.So Xxx can be replaced by any return type.
   Program to demonstrate parseXxx()::

```
class Wrap2
{
public static void main(String[] args)
{
Integer i=Integer.parseInt("123");
System.out.println(i);
}
}
```

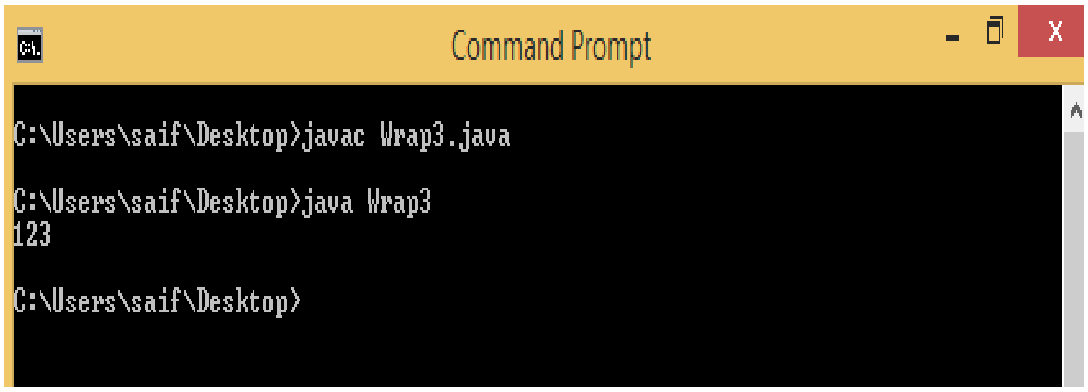
   Output:



```
C:\Users\saif\Desktop>javac Wrap2.java

C:\Users\saif\Desktop>java Wrap2
123

C:\Users\saif\Desktop>_
```

3. **XxxValue():** It is an instance method of the wrapper class and Xxx can be replaced by the any primitive data type and return to corresponding data type. Since it is an instance method so we used dot operator in it.
   Program to demonstrate:

```
class Wrap3
{
public static void main(String[] args)
{
Integer x=Integer.valueOf("123");
int a=x.intValue();
System.out.println(a);
}
}
```



```
C:\Users\saif\Desktop>javac Wrap3.java

C:\Users\saif\Desktop>java Wrap3
123

C:\Users\saif\Desktop>
```

Abstract class:

An abstract class is used to hide the functionality detail of the class. It lets you to focus on what your program does not in how your program does your work.

For Example: In sending a sms from one source to destination you always focus on your message not on how message goes from sender to receiver.

Properties of the abstract class:

1. Java abstract classes can be used to represent the common characteristic of the sub class.
2. This class can only be the parent class where the sub class will extends it.
3. It is like any other normal class which contains the variable, method and constructor etc with a keyword abstract before the class name.
4. You cannot create the object of abstract class but you can create the reference variable of the abstract class.
5. If you have any method which is abstract then it is mandatory for you to define the class also as abstract.
6. A class can be declaring abstract without an abstract method but an abstract method cannot be declared without an abstract class.

Syntax for abstract class:

Abstract class Saif()

```
{
……….
………..
}
```

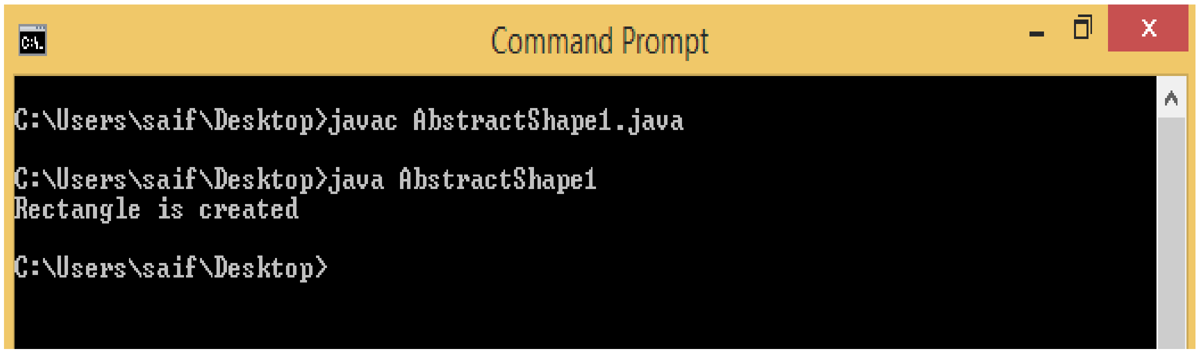
Syntax for abstract method

```
Abstract class Saif()
{
Abstract  void disp();
Void disp()
{
….
}
}
```

Program of Abstract class with abstract method:

```
abstract class AbstractShape
{
abstract void run();
}
class Rectangle extends AbstractShape
{
void run()
{
System.out.println("Rectangle is created");
}
}
class Circle extends AbstractShape
{
void run()
{
System.out.println("Cirle is created");
}
}
class AbstractShape1
{
public static void main(String[] args)
{
AbstractShape a=new Circle();
a.run();
}
}
```

Output:



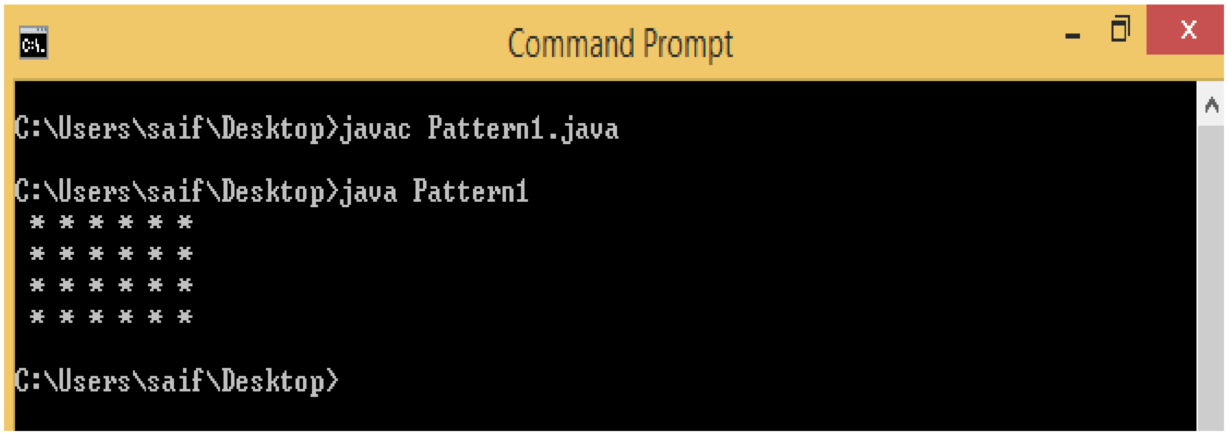
```
C:\Users\saif\Desktop>javac AbstractShape1.java

C:\Users\saif\Desktop>java AbstractShape1
Rectangle is created

C:\Users\saif\Desktop>
```

Program of Pattern:

The pattern program is generally done by the loop of i,j where I is for row line and j is for the coloumn.

1. **Program to print the following pattern:**
```
******
******
******
******
```

```
class Pattern1
{
public static void main(String[] args)
{
int i,j;
for(i=1;i<=4;i++)
{
for(j=1;j<=6;j++)
{
System.out.print(" ");
System.out.print("*");
}
System.out.println("");
}
}
}
```

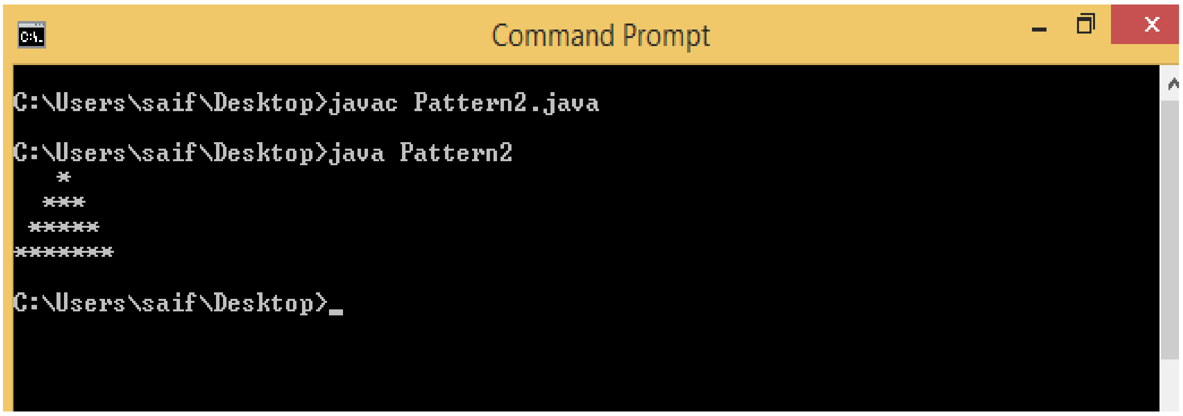


```
C:\Users\saif\Desktop>javac Pattern1.java

C:\Users\saif\Desktop>java Pattern1
 * * * * * *
 * * * * * *
 * * * * * *
 * * * * * *

C:\Users\saif\Desktop>
```

Program 2: Print the following pattern

```
class Pattern2
{
public static void main(String[] args)
{
int i,j,k;
for(i=1;i<=4;i++)
{
for(j=4;j>i;j--)
{
System.out.print(" ");
}
for(k=1;k<=(2*i-1);k++)
{
System.out.print("*");
}
System.out.println("");
}
}
}
```

Output:

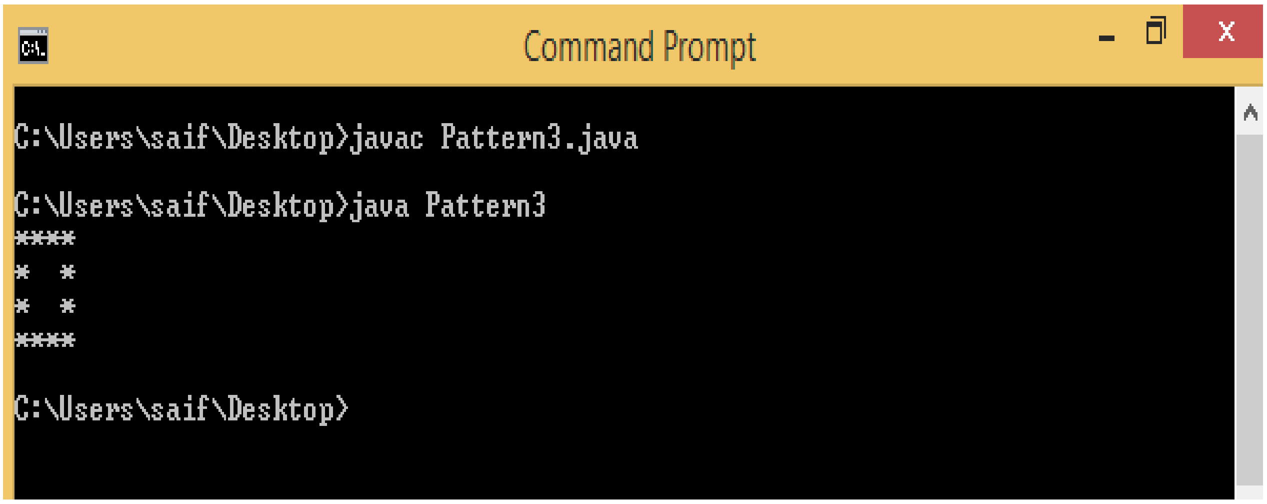

Command Prompt

```
C:\Users\saif\Desktop>javac Pattern2.java

C:\Users\saif\Desktop>java Pattern2
   *
  ***
 *****
*******

C:\Users\saif\Desktop>_
```

Program: Print the following Pattern:

```
class Pattern3
{
public static void main(String[] args)
{
int i,j;
for(i=1;i<=4;i++)
  {
   for(j=1;j<=4;j++)
       {
       if(i==1 || i==4 || j==1 || j==4)
         {
         System.out.print("*");
         }
        else
         {
         System.out.print(" ");
         }
        }
        System.out.println("");
    }
}
}
```

Output:



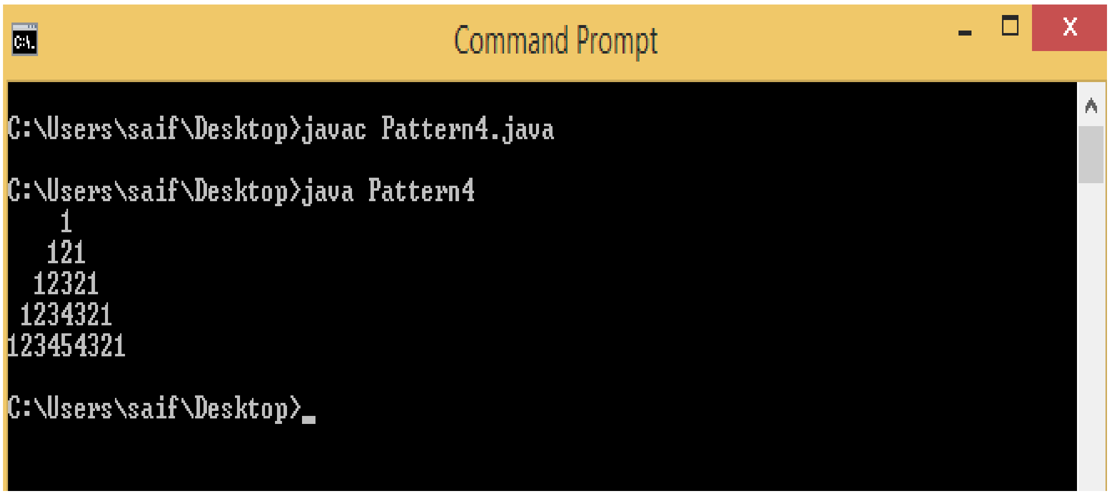
```
C:\Users\saif\Desktop>javac Pattern3.java

C:\Users\saif\Desktop>java Pattern3
****
*  *
*  *
****

C:\Users\saif\Desktop>
```

Print the following pattern:

```
class Pattern4
{
 public static void main(String[] args)
   {
    int odd=1;
    int ns=3;
      for(int i=1;i<=5;i++)
          {
           int k=0;
           for(int j=0;j<=ns;j++)
               {
               System.out.print(" ");
```

```
                }
           for(int j=1;j<=odd;j++)
          {
           if(j<=i)
                {
                k=k+1;
                 }
           else
                {
                k=k-1;
                 }
       System.out.print(k);
       }
System.out.println("");
odd=odd+2;
ns=ns-1;
}
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Pattern4.java

C:\Users\saif\Desktop>java Pattern4
    1
   121
  12321
 1234321
123454321

C:\Users\saif\Desktop>_
```

Interface:

Interface in java means that you are signing a contract to achieve all the method of the class.

The first party of the contract is the user(Generally Interface) and second pparty of the contract is the class. When a class implement the interface then it inherit all the method of the class user.

The interface is achieve by the keyword interface while the class can extends all the method of the class with the help of the keyword implements.

Syntax:

```
interface A
{
void  a();
}
class B implements A
{
void a()
{
}
Public staticvoid main(String[] args)
B b=new B();
b.a();
}
```

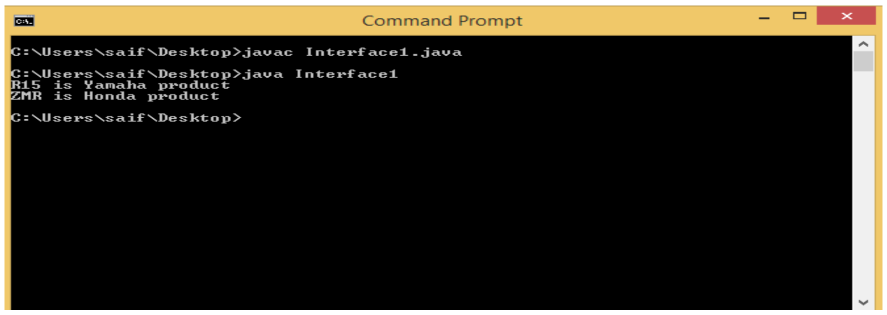
About Program:

In the program A is a interface which have the method name a().We do not define the method in this class as it is the property of the interface class which is abstract. The new class B will implement the method of the interface A and its method can be achieved by its object b.

Note: All the method in the class are public and abstract while all the data members in the class are public static and final keyword. When the program is compiler then the compiler will automatically make the method as the public and abstract by default and the data members will automatically create as the private static and final.

There are following use of the Interface in Java:

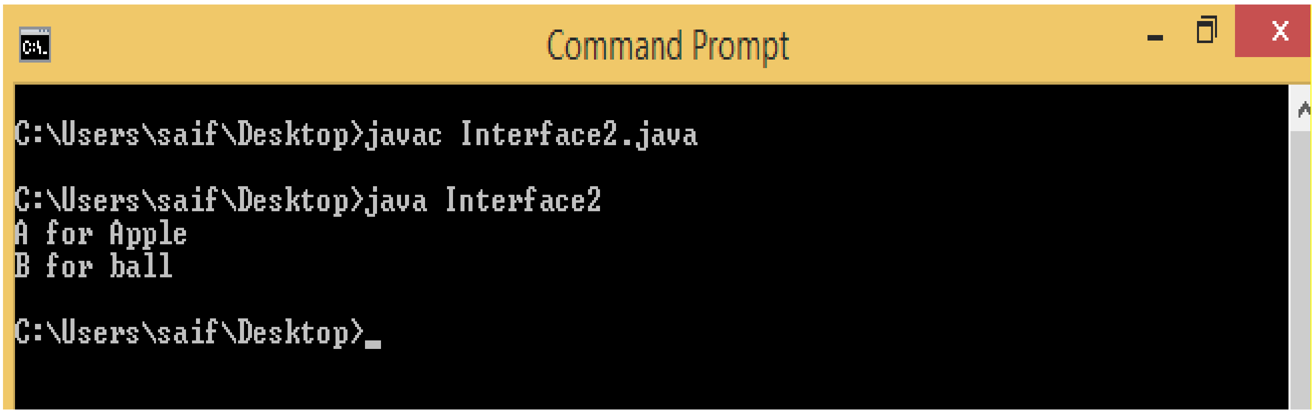
1. The most important use of the interface is to achieve the multiple inheritances. As we all know that in Java multiple inheritance is not supported due to its theory of simplicity but we have interface is the way which allow us to achieve multiple inheritance.
   Program on multiple inheritance using interface
   ```
   interface R15
   {
   void run1();
   }
   interface ZMR
   {
   void run2();
   }
   class Interface1 implements R15, ZMR
   {
   public void run1()
   {
   System.out.println("R15 is Yamaha product");
   }
   public void run2()
   {
   System.out.println("ZMR is Honda product");
   }
   public static void main(String[] args)
   {
   Interface1 i=new Interface1();
   i.run1();
   i.run2();
   }
   }
   ```



```
C:\Users\saif\Desktop>javac Interface1.java

C:\Users\saif\Desktop>java Interface1
R15 is Yamaha product
ZMR is Honda product

C:\Users\saif\Desktop>
```

Program on how 1 interface class can extends the another interface class and a object class can implements the B and achieved the abstract method of class A as well.

```
interface A
{
void a();
}
interface B extends A
{
void b();
}
class Interface2 implements B
{
   public void a()
        {
         System.out.println("A for Apple");
        }
    public void b()
        {
        System.out.println("B for ball");
        }
 public static void main(String[] args)
    {
    Interface2 c=new Interface2();
    c.a();
    c.b();
  }
}
```



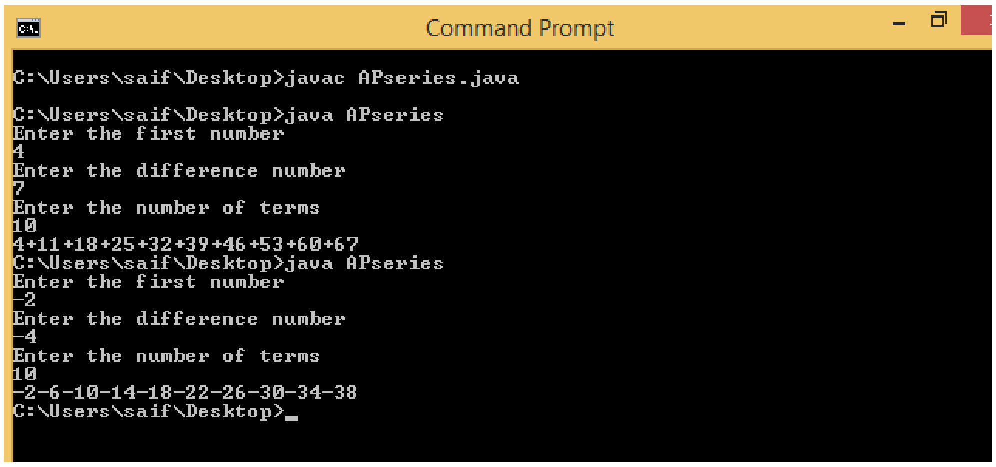
```
C:\Users\saif\Desktop>javac Interface2.java

C:\Users\saif\Desktop>java Interface2
A for Apple
B for ball

C:\Users\saif\Desktop>_
```

Program: Print an AP series in java

```
import java.util.*;
class APseries
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the first number");
int a=in.nextInt();
System.out.println("Enter the difference number");
int d=in.nextInt();
System.out.println("Enter the number of terms");
int n=in.nextInt();
for(int i=1;i<=n;i++)
{
 if(i<n)
```

```
   {
      if(a>=0 && i>1)
        {

          System.out.print("+"+a);
        }
      else
        {
          System.out.print(a);
        }
   }
 else
        {
          if(a>0)
          {
        System.out.print("+"+a);
           }
          else
           {
          System.out.print(+a);
            }
        }
 a=a+d;
}
}
}
```

Output:

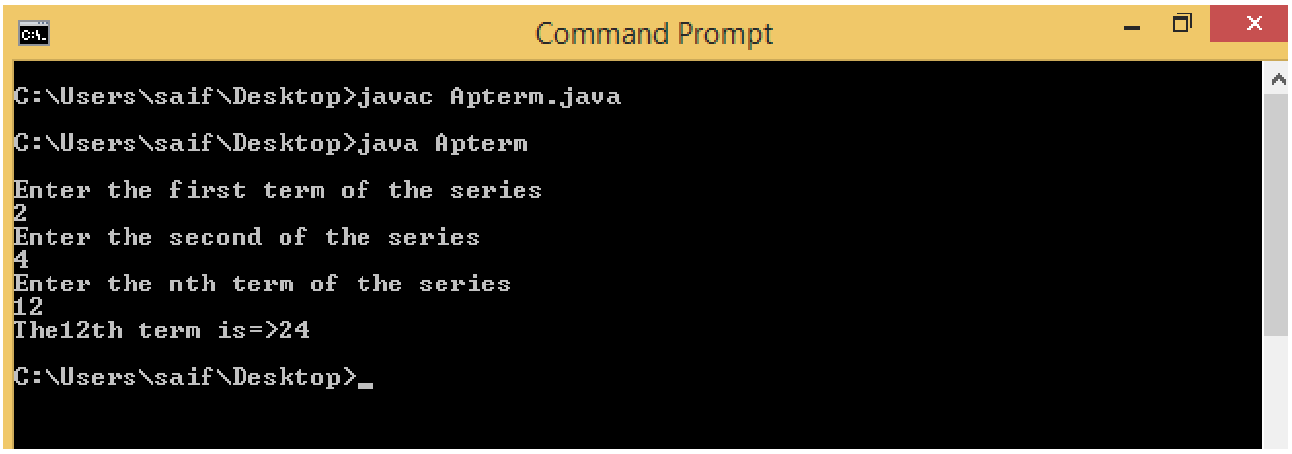


```
C:\Users\saif\Desktop>javac APseries.java

C:\Users\saif\Desktop>java APseries
Enter the first number
4
Enter the difference number
7
Enter the number of terms
10
4+11+18+25+32+39+46+53+60+67
C:\Users\saif\Desktop>java APseries
Enter the first number
-2
Enter the difference number
-4
Enter the number of terms
10
-2-6-10-14-18-22-26-30-34-38
C:\Users\saif\Desktop>_
```

Program to find nth term of the series in Java

```
import java.util.*;

class Apterm

{

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);

System.out.println();

System.out.println("Enter the first term of the series");

int a=in.nextInt();

System.out.println("Enter the second of the series");

int b=in.nextInt();

int d=b-a;

System.out.println("Enter the nth term of the series");

int n=in.nextInt();

int t=a+(n-1)*d;

System.out.println("The"+n+"th term is=>"+t);

}

}
```

Output:



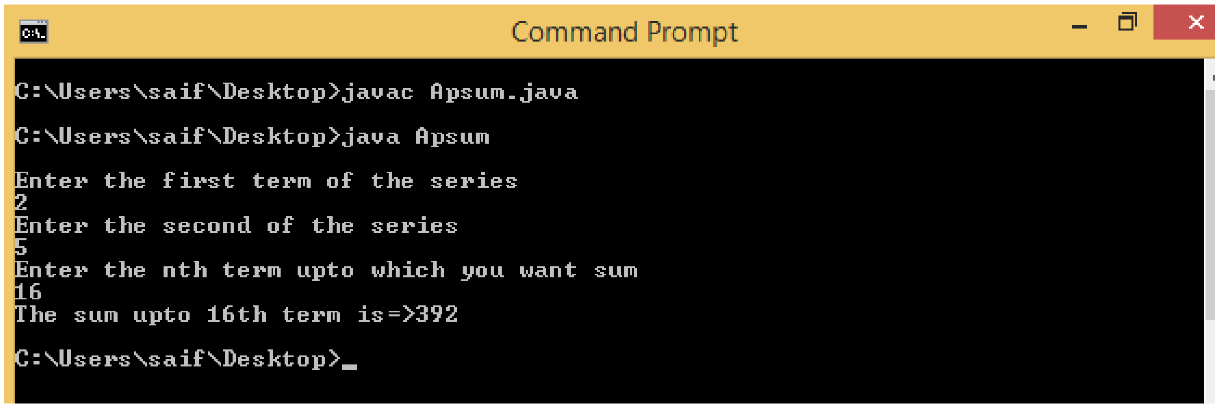
```
C:\Users\saif\Desktop>javac Apterm.java

C:\Users\saif\Desktop>java Apterm

Enter the first term of the series
2
Enter the second of the series
4
Enter the nth term of the series
12
The12th term is=>24

C:\Users\saif\Desktop>_
```

Program to find the sum of the series is

```
import java.util.*;

class Apsum

{

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);

System.out.println();

System.out.println("Enter the first term of the series");

int a=in.nextInt();

System.out.println("Enter the second of the series");

int b=in.nextInt();

int d=b-a;

System.out.println("Enter the nth term upto which you want sum");

int n=in.nextInt();

int s=n/2*(2*a+(n-1)*d);

System.out.println("The sum upto "+n+"th term is=>"+s);

}
```

}

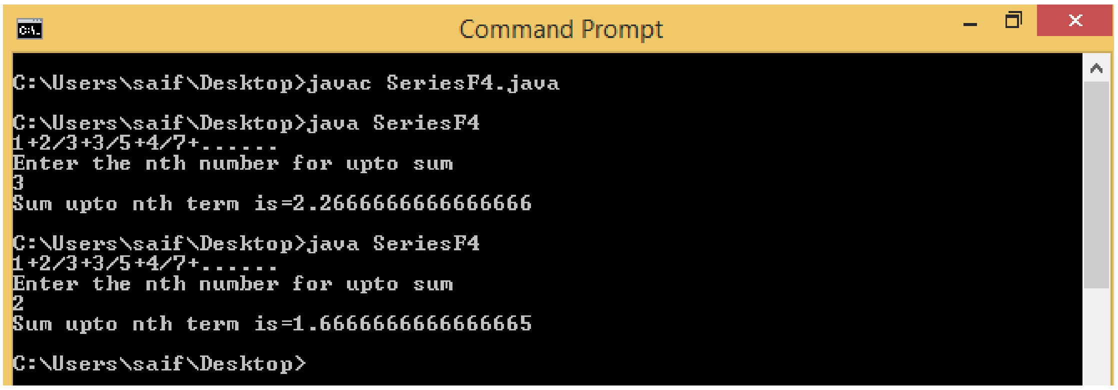
Output



```
C:\Users\saif\Desktop>javac Apsum.java

C:\Users\saif\Desktop>java Apsum

Enter the first term of the series
2
Enter the second of the series
5
Enter the nth term upto which you want sum
16
The sum upto 16th term is=>392

C:\Users\saif\Desktop>_
```

Program to find the nth term of the series:

```
import java.util.*;

class SeriesF4

{

public static void main(String[] args)

{

double sum=0;

int odd=1;

Scanner in=new Scanner(System.in);

System.out.println("1+2/3+3/5+4/7+......");

System.out.println("Enter the nth number for upto sum");

int n=in.nextInt();

for(int i=1;i<=n;i++)

{

sum=sum+(double)i/odd;

odd=odd+2;

}
System.out.println("Sum upto nth term is="+sum);
}
}
```

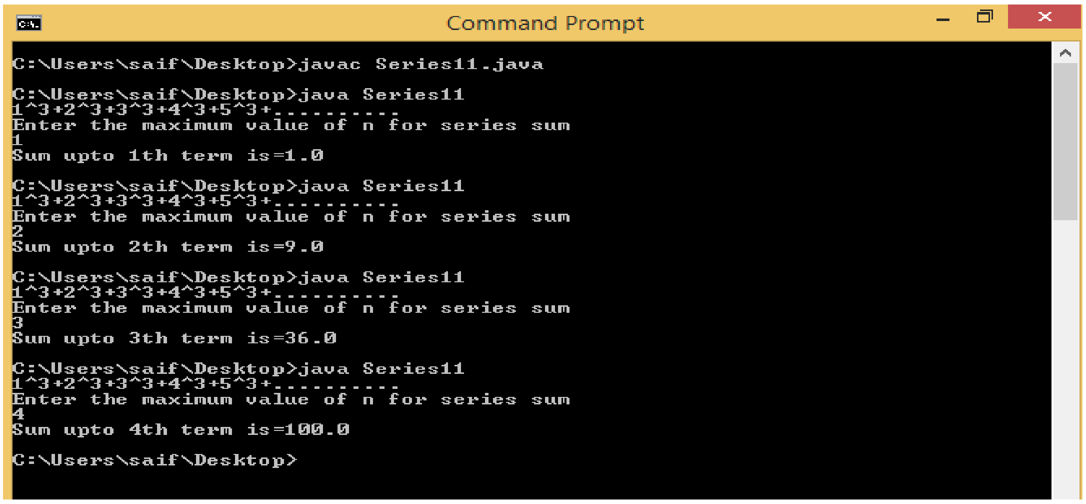


```
C:\Users\saif\Desktop>javac SeriesF4.java

C:\Users\saif\Desktop>java SeriesF4
1+2/3+3/5+4/7+......
Enter the nth number for upto sum
3
Sum upto nth term is=2.2666666666666666

C:\Users\saif\Desktop>java SeriesF4
1+2/3+3/5+4/7+......
Enter the nth number for upto sum
2
Sum upto nth term is=1.6666666666666665

C:\Users\saif\Desktop>
```

Program to find the series 1^3+2^3+3^3+5^3…….sum up to nth term

```
import  java.util.*;
class Series11
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("1^3+2^3+3^3+4^3+5^3+..........");
System.out.println("Enter the maximum value of n for series sum");
int n=in.nextInt();
double sum=0;
for(int i=1;i<=n;i++)
{
sum=sum+(Math.pow(i,3));
}
System.out.println("Sum upto "+n+"th term is="+sum);
}
}
```

Output:



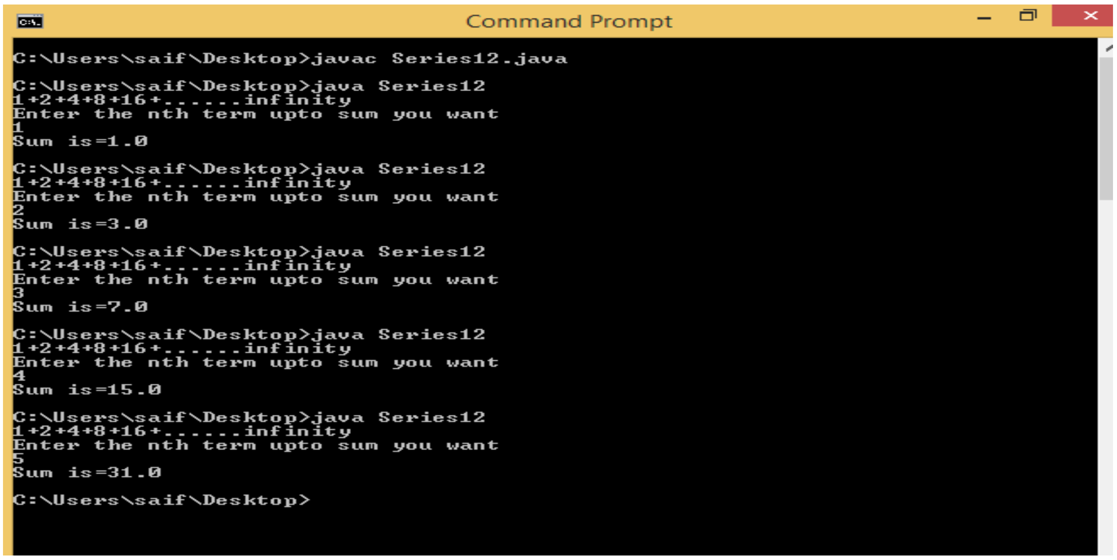
```
C:\Users\saif\Desktop>javac Series11.java

C:\Users\saif\Desktop>java Series11
1^3+2^3+3^3+4^3+5^3+..........
Enter the maximum value of n for series sum
1
Sum upto 1th term is=1.0

C:\Users\saif\Desktop>java Series11
1^3+2^3+3^3+4^3+5^3+..........
Enter the maximum value of n for series sum
2
Sum upto 2th term is=9.0

C:\Users\saif\Desktop>java Series11
1^3+2^3+3^3+4^3+5^3+..........
Enter the maximum value of n for series sum
3
Sum upto 3th term is=36.0

C:\Users\saif\Desktop>java Series11
1^3+2^3+3^3+4^3+5^3+..........
Enter the maximum value of n for series sum
4
Sum upto 4th term is=100.0

C:\Users\saif\Desktop>
```

Program to find the sum of series 1+2+4+8+16+…….infinity

```
import java.util.*;
class Series12
{
public static void main(String[] args)
{
  double sum=0;
  Scanner in=new Scanner(System.in);
  System.out.println("1+2+4+8+16+......infinity");
  System.out.println("Enter the nth term upto sum you want");
  int n=in.nextInt();
  for(int i=0;i<n;i++)
    {
     sum=sum+(Math.pow(2,i));
    }
  System.out.println("Sum is="+sum);
}
```

}

Output



```
C:\Users\saif\Desktop>javac Series12.java

C:\Users\saif\Desktop>java Series12
1+2+4+8+16+......infinity
Enter the nth term upto sum you want
1
Sum is=1.0

C:\Users\saif\Desktop>java Series12
1+2+4+8+16+......infinity
Enter the nth term upto sum you want
2
Sum is=3.0

C:\Users\saif\Desktop>java Series12
1+2+4+8+16+......infinity
Enter the nth term upto sum you want
3
Sum is=7.0

C:\Users\saif\Desktop>java Series12
1+2+4+8+16+......infinity
Enter the nth term upto sum you want
4
Sum is=15.0

C:\Users\saif\Desktop>java Series12
1+2+4+8+16+......infinity
Enter the nth term upto sum you want
5
Sum is=31.0

C:\Users\saif\Desktop>
```

Exception handling:

As we all know that exception means Abnormal. So in java to handle the abnormal behavior of the program we use Exception Handling. This is the powerful mechanism to control the normal flow of program.

In exception handling we generally handle the run time error in the program such as divide value by zero, input output error, class not found , SQL , Remote etc.

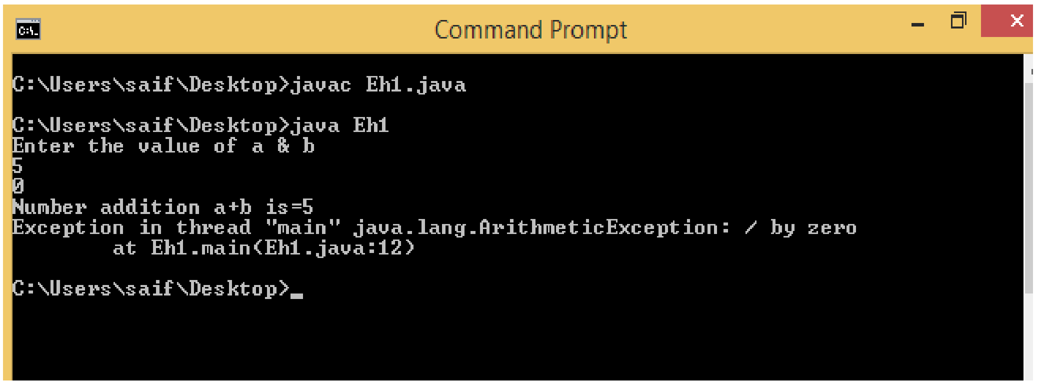
During exception in the program the JVM will not be able to take any decision and execution gets totally stopped.

Exception only occur at run time they never occur at compile time.

Program of exception situation in java

```
import java.util.*;

class Eh1

{

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);
System.out.println("Enter the value of a & b");
int a=in.nextInt();
int b=in.nextInt();
int c=a+b;
System.out.println("Number addition a+b is="+c);
int d=a/b;
System.out.println("Number addition a+b is="+d);
int e=a*b;
System.out.println("Number addition a+b is="+e);
}
}
```

Output:



```
C:\Users\saif\Desktop>javac Eh1.java

C:\Users\saif\Desktop>java Eh1
Enter the value of a & b
5
0
Number addition a+b is=5
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Eh1.main(Eh1.java:12)

C:\Users\saif\Desktop>_
```

Program start from normal execution and add the value of a and b at statement 1. but as soon as it divide by 0 the program terminate and throw the exception from statement 2 and terminate the whole program without executing the statement 3.

Following are the advantage of exception handling in Java:

1. It separate the error code from the whole program
2. It group the error with a special message
3. It allows the compiler to complete the code until it end.
4. It prevent the compiler from being terminate the code after an error occur.

Example:

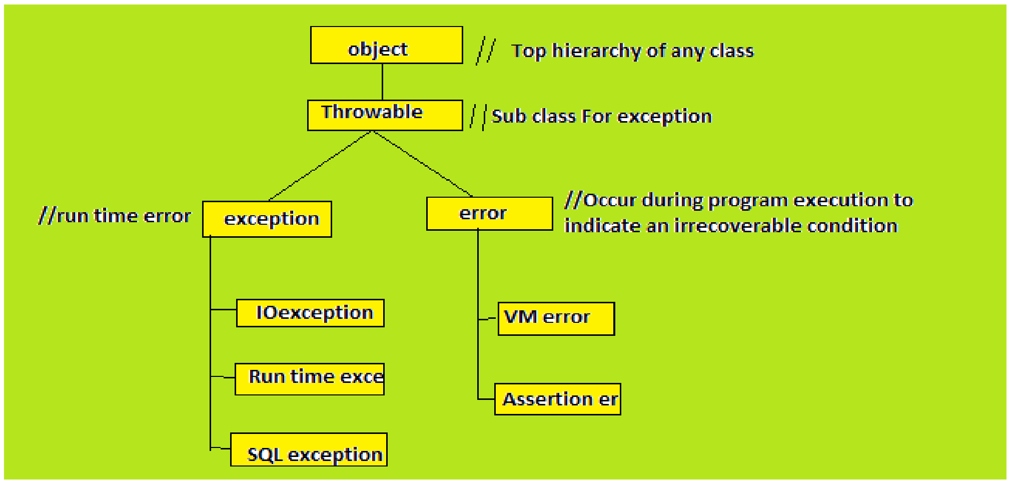
Suppose you have a program with 8 statements as given below:

1. statement 1;
2. statement 2;
3. statement 3;
4. statement 4; //exception occurs
5. statement 5 ;
6. statement 6;
7. statement 7;
8. statement 8;

Without exception handling the program will terminate from statement 4 and exit from the program. So normal execution of the program will completely disrupt.

Now if we used the exception handling in our program the program will catch the exception and throw it to the object and complete the other statement and will not exit until the statement 8 executed

Exception hierarchy in Java


object
// Top hierarchy of any class
Throwable
//Sub class For exception
//run time error
exception
error
//Occur during program execution to indicate an irrecoverable condition
IOexception
Run time exce
SQL exception
VM error
Assertion er

Types of exception in Java

According to the sun microsystem there are three types of exception in Java:

1. Checked Exception
2. Unchecked Exception
3. Error

Checked Exception:

Checked exception are the compile time exception such as IOExeption , SQLException , ClassNotFoundException etc. Java is the first kanguage that introduce the checked exception from JDK1.

Unchecked Exception: Unchecked exception are the run time exception Such as ArrayIndexOutofBond , NullPointerException etc which are the part of the java.lang.RuntimeException class.The unchecked exception is not checked by the java compiler but it is run time exception due to bad code in your program.

Error

An error is a serious problem that a reasonable application shoud not try to catch.It is the part of java.lang.error class for example java.lang.StackOverflowError, java.lang.OutOfMemoryError etc.

Difference between error and exception:

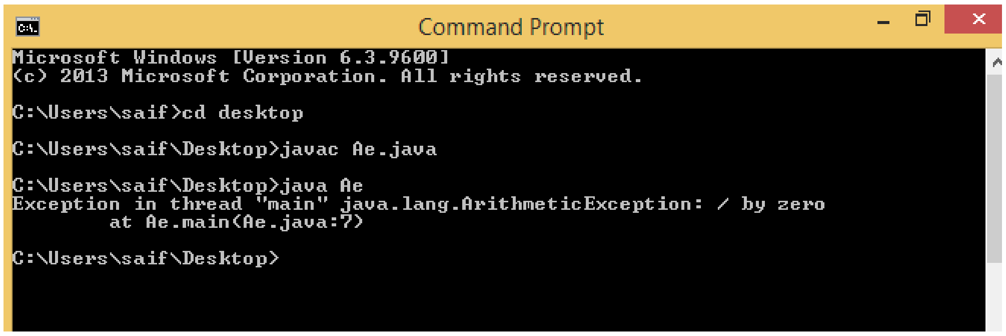
Error	**Exception**
1. Error in Java are unchecked type	Exception are both type checked and unchecked
2. It is impossible to recover from the error.	Through try and catch block we can easily recover from the exception.
3. It is mostly caused by the environment in which program is running	It is mostly cause by the application of the program..
4. It is the part of the java.lang.Error class	It is the part of the java.lang.Exception
5. It is unknown by the compiler	Checked exception are known by the compiler where as unchecked exception are not known by compiler.

Example of the common exception:

1. **Arithmetic Exception:** If a number is divide by the zero in the program then there will occur arithmetic exception.

```
class Ae
{
public static void main(String[] atgs)
{
int a=5;
int b=0;
System.out.println("Division is="+a/b);
}
}
```

Output:

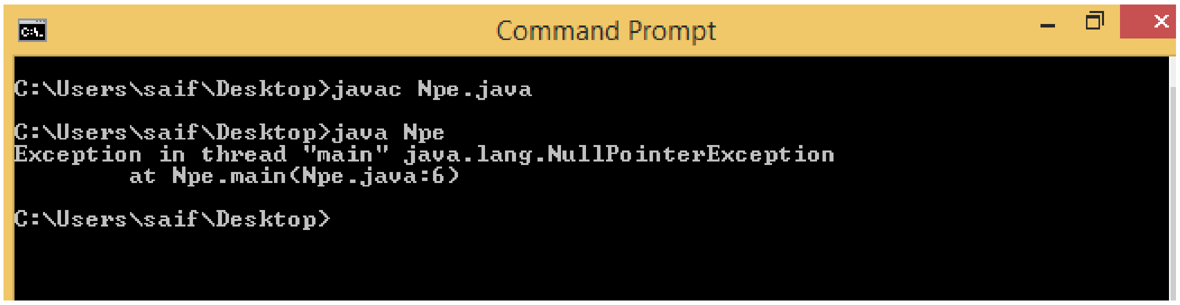


```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\saif>cd desktop

C:\Users\saif\Desktop>javac Ae.java

C:\Users\saif\Desktop>java Ae
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Ae.main(Ae.java:7)

C:\Users\saif\Desktop>
```

2. **NullPointerException:** NullPointerException occur when you have variable and you want to perform some operation on it.if your variable contain a null value then there will occur NullPoiterException due to null value and performing operation on it.

```
class Npe
{
public static void main(String[] args)
{
String a=null;
System.out.println(a.length());
}
}
```

Output:

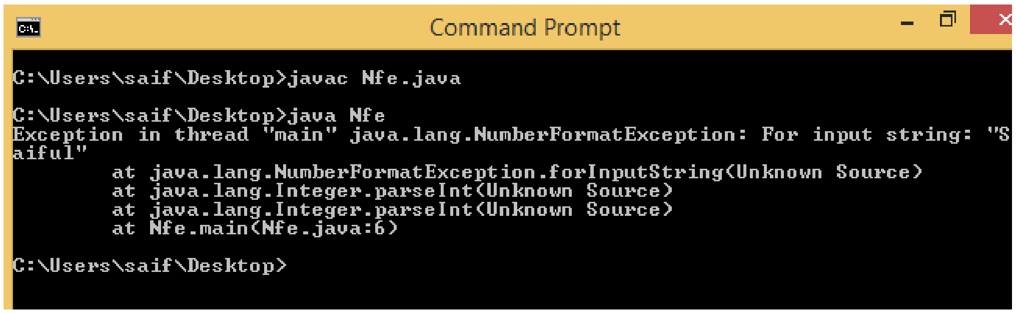


```
C:\Users\saif\Desktop>javac Npe.java

C:\Users\saif\Desktop>java Npe
Exception in thread "main" java.lang.NullPointerException
        at Npe.main(Npe.java:6)

C:\Users\saif\Desktop>
```

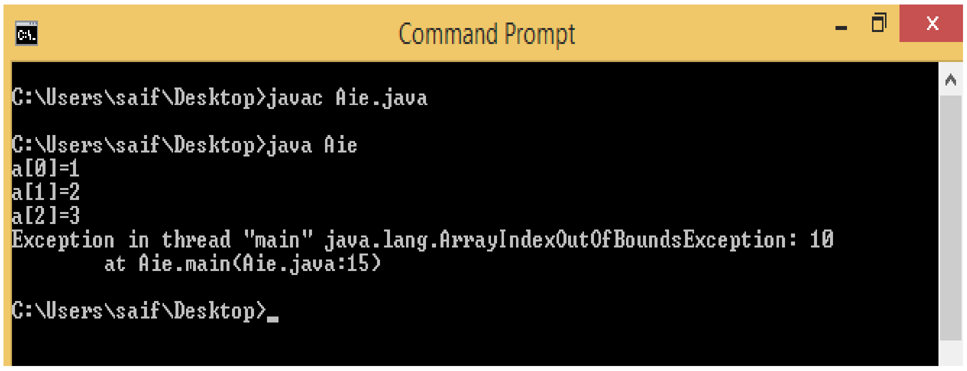
3. **NumberFormatException:** Number format exception occur when you try to convert an string into number type like int, float, double etc.

```
class Nfe
{
public static void main(String[] args)
{
String s="Saiful";
int i=Integer.parseInt(s);
System.out.println(i);
}
}
```

Output:


```
C:\Users\saif\Desktop>javac Nfe.java

C:\Users\saif\Desktop>java Nfe
Exception in thread "main" java.lang.NumberFormatException: For input string: "S
aiful"
        at java.lang.NumberFormatException.forInputString(Unknown Source)
        at java.lang.Integer.parseInt(Unknown Source)
        at java.lang.Integer.parseInt(Unknown Source)
        at Nfe.main(Nfe.java:6)

C:\Users\saif\Desktop>
```

4. **ArrayIndexOutOfBoundsException:** This exception occur when we perform the operation out of the array memory position.
Program:
```
class Aie
{
public static void main(String[] args)
{
int a[]=new int[5];
a[0]=1;
a[1]=2;
a[2]=3;
a[3]=4;
a[4]=5;
System.out.println("a[0]="+a[0]);
System.out.println("a[1]="+a[1]);
System.out.println("a[2]="+a[2]);
System.out.println("a[10]="+a[10]);
}
}
```


```
C:\Users\saif\Desktop>javac Aie.java

C:\Users\saif\Desktop>java Aie
a[0]=1
a[1]=2
a[2]=3
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
        at Aie.main(Aie.java:15)

C:\Users\saif\Desktop>_
```

Keyword for the exception handling:

1. try
2. catch
3. finally
4. throw
5. throws

Try-catch: We know that exception terminate the whole program without end it up. So in a code of there is a situation for any kind of exception then we use try block so exception cannot end the program.

If there is a chance for a exception in the code then we enclose that code in try block so that program should not end due to the exception.

The try block is always followed either by the catch or finally block.

Real world Example:

In the above picture you are trying to walk on the tight rope between cliffs of the two mountains. Here we use try because we are not sure whether we success or not. What will happen if you slip from the rope? You will definitely die and your life will end.

But if there is a catch at the bottom then you will save and can continue your life.

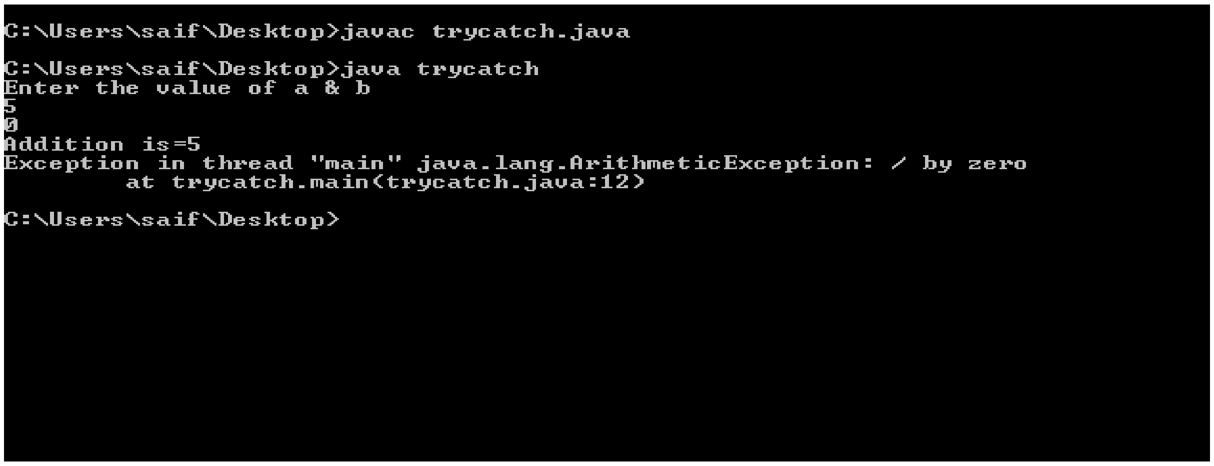
In the same way if there is a block without try catch statement then if exception arises code will end but if there is try catch then the code will execute until it reaches its end.

Program without try-catch block

```
import java.util.*;

class trycatch

{
```

```
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the value of a & b");
int a=in.nextInt();
int b=in.nextInt();
int c=a+b;
System.out.println("Addition is="+c);
int d=a/b;
System.out.println("Addition is="+d);
int e=a*b;
System.out.println("Addition is="+e);
}
}
```

Output:



```
C:\Users\saif\Desktop>javac trycatch.java

C:\Users\saif\Desktop>java trycatch
Enter the value of a & b
5
0
Addition is=5
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at trycatch.main(trycatch.java:12)

C:\Users\saif\Desktop>
```

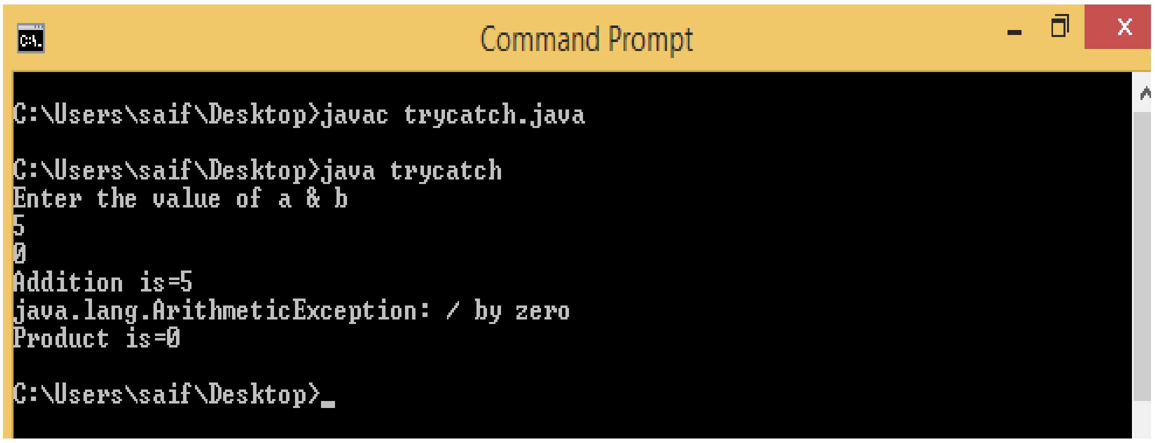
Program add the variable but throw an exception when divide by zero without continuing the third statement.

Program with try-catch block for exception handling

```
import java.util.*;
class trycatch
```

```
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the value of a & b");
int a=in.nextInt();
int b=in.nextInt();
int c=a+b;
System.out.println("Addition is="+c);
try
{
int d=a/b;
System.out.println("Addition is="+d);
}
catch(ArithmeticException z)
{
System.out.println(z);
}
int e=a*b;
System.out.println("Product is="+e);
}
}
```

Output



```
C:\Users\saif\Desktop>javac trycatch.java

C:\Users\saif\Desktop>java trycatch
Enter the value of a & b
5
0
Addition is=5
java.lang.ArithmeticException: / by zero
Product is=0

C:\Users\saif\Desktop>_
```

So the conclusion is that try block help us to catch the exception without terminate the code.

Multiple catch statement:

In java we can use multiple exceptions for same to perform different task at the occurrence of the different exception.

In java at a time only one exception can occurred and then at that time only one catch block can then executed.

There is a rule for writing multiple catch statement in a program. The ordered of the catch block must be most specific to most general. For example ArithmaticException must be come first before the ArrayIndexOutOfBond exceptions in the catch statement. If we give the priority to the second over first then we will get compile time error.

Nested try block:

With the help of the nested try block we can use the try block into another try block. Some times in a program there is problem arises that part of the program cause one exception and the part of the program cause another exception while whole program may cause another exception. So in this case we can use the nested try statement.

Program to demonstrate Nested Try statement

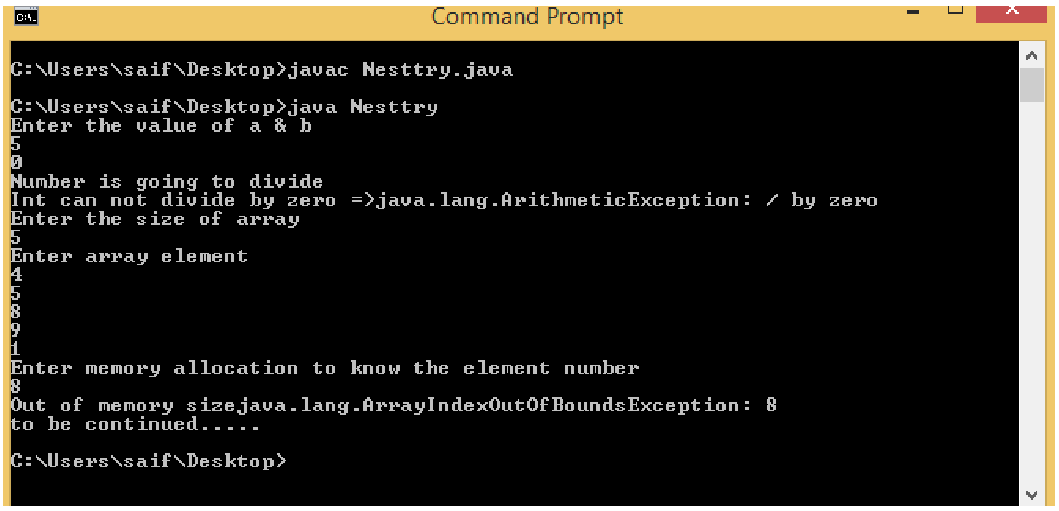
```
import java.util.*;

class Nesttry

{

public static void main(String[] args)

{

  Scanner in=new Scanner(System.in);;

  System.out.println("Enter the value of a & b");
```

```
 int a=in.nextInt();
 int b=in.nextInt();
 try
  {
   System.out.println("Number is going to divide");
   try
    {
     int c=a/b;
     System.out.println("Answer is="+c);
    }
   catch(ArithmeticException e)
    {
     System.out.println("Int can not divide by zero =>"+e);
    }
   try
    {
    int x[]=new int[5];
    System.out.println("Enter the size of array");
    int m=in.nextInt();
    System.out.println("Enter array element");
    for(int i=0;i<m;i++)
    {
     x[i]=in.nextInt();
     }
    System.out.println("Enter memory allocation to know the element number");
    int n=in.nextInt();
    System.out.println("at x["+n+"] number enter by you is"+x[n]);
   System.out.println("Successfully alot array memory location");
```

```
    }
    catch(ArrayIndexOutOfBoundsException e)
    {
   System.out.println("Out of memory size"+e);
    }
  }
  catch(Exception e)
    {
    System.out.println("Exception found");
    }
System.out.println("to be continued.....");
}
}
```

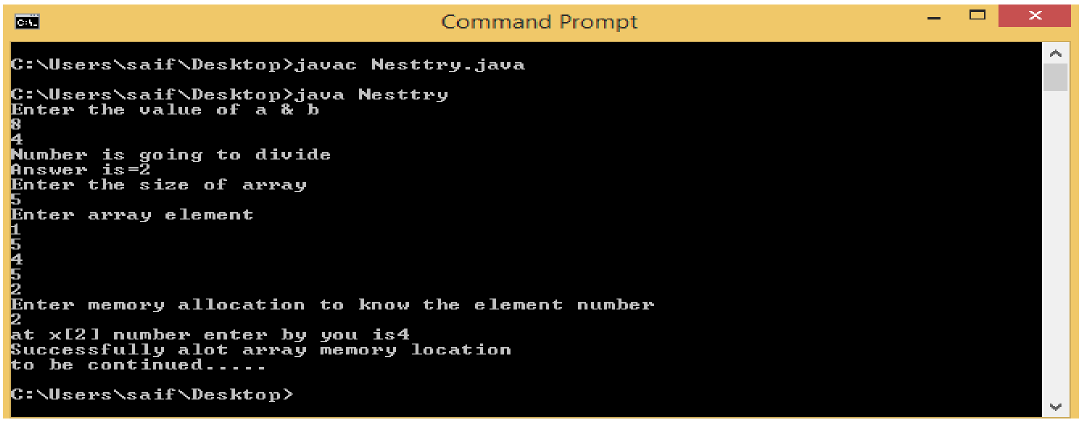
Output:

With exception arises but program is running



```
C:\Users\saif\Desktop>javac Nesttry.java

C:\Users\saif\Desktop>java Nesttry
Enter the value of a & b
5
0
Number is going to divide
Int can not divide by zero =>java.lang.ArithmeticException: / by zero
Enter the size of array
5
Enter array element
4
5
8
9
1
Enter memory allocation to know the element number
8
Out of memory sizejava.lang.ArrayIndexOutOfBoundsException: 8
to be continued.....

C:\Users\saif\Desktop>
```

Without exception



```
C:\Users\saif\Desktop>javac Nesttry.java

C:\Users\saif\Desktop>java Nesttry
Enter the value of a & b
8
4
Number is going to divide
Answer is=2
Enter the size of array
5
Enter array element
1
5
4
5
2
Enter memory allocation to know the element number
2
at x[2] number enter by you is4
Successfully alot array memory location
to be continued.....

C:\Users\saif\Desktop>
```

Finally block:

Java finally block is always executed after the try-catch block whether the exception is handle or not in the program. After all the processing done by the try catch statement then at last finally block is executed.

If there is no exception in the program then finally statement executed after the try block but in case if exception is arises then finally block executed after the catch block.

If you use System.exit(); in the statement then finally block will not execute and the program terminate from the try block otherwise finally block is always execute whether it is declare or not.

There can be more than one catch block with a try block but there exist only one try block.

Use of the finally block

Finally block is used to clean up some unmanaged resource.It is the method to clean the some code at one place.

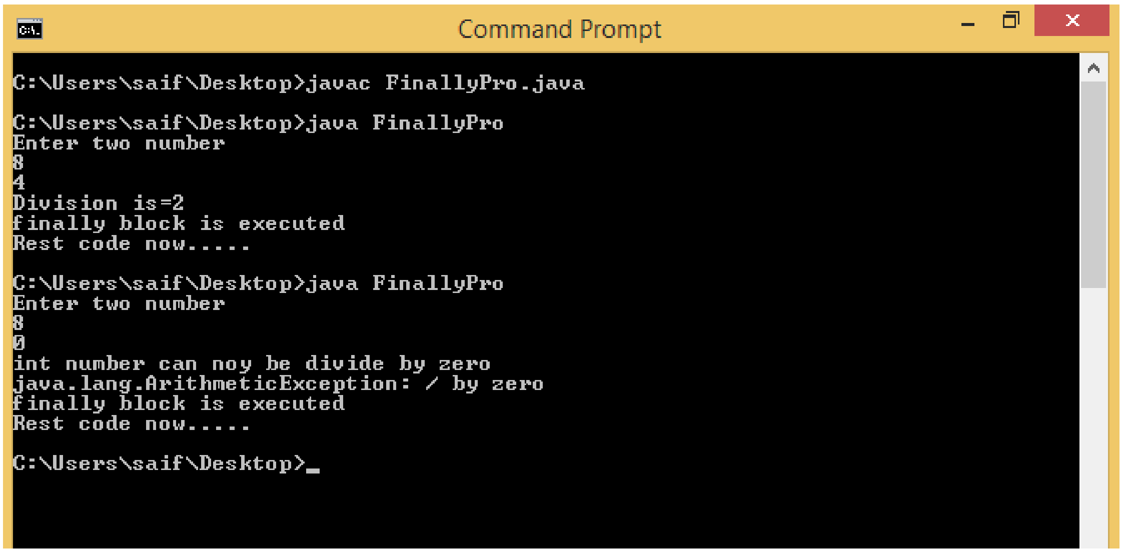
For example your application is closed forcefully but there is some task which you really want to execute. So if you write these lines of code in the finally block then it will execute whether the exception is thrown or not.

But in case of JVM fail and thread terminate then JVM will not execute.

Program to demonstrate the finally keyword in java:

```
import java.util.*;

class FinallyPro

{

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);
System.out.println("Enter two number");
int a=in.nextInt();
int b=in.nextInt();
try
{
int d=a/b;
System.out.println("Division is="+d);
}
catch(ArithmeticException e)
{
System.out.println("int number can noy be divide by zero");
System.out.println(e);
}
finally
{
System.out.println("finally block is executed");
}
System.out.println("Rest code now.....");
}
}
```

Output:



```
C:\Users\saif\Desktop>javac FinallyPro.java

C:\Users\saif\Desktop>java FinallyPro
Enter two number
8
4
Division is=2
finally block is executed
Rest code now.....

C:\Users\saif\Desktop>java FinallyPro
Enter two number
8
0
int number can noy be divide by zero
java.lang.ArithmeticException: / by zero
finally block is executed
Rest code now.....

C:\Users\saif\Desktop>_
```

throw:

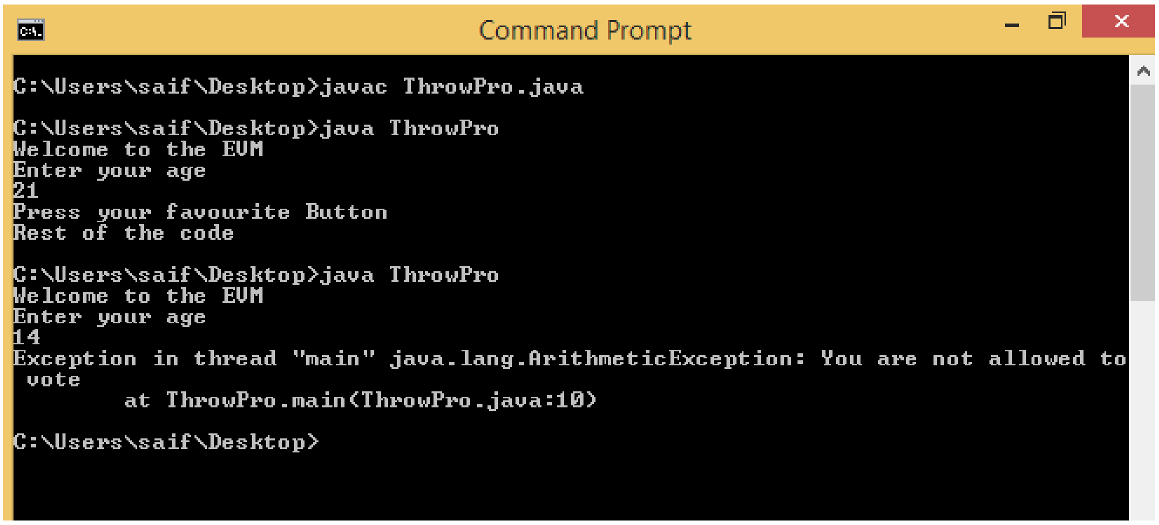
To explicitly throw an exception we use the throw keyword in java. With the help of this keyword we can throw a checked and unchecked exception in java.

Program to demonstrate the throw keyword

```
import java.util.*;

class ThrowPro

{

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);

System.out.println("Welcome to the EVM");

System.out.println("Enter your age");

int n=in.nextInt();

if(n<18) throw new ArithmeticException("You are not allowed to vote");

else

System.out.println("Press your favourite Button");

System.out.println("Rest of the code");

}
```

}

Output:



```
C:\Users\saif\Desktop>javac ThrowPro.java

C:\Users\saif\Desktop>java ThrowPro
Welcome to the EVM
Enter your age
21
Press your favourite Button
Rest of the code

C:\Users\saif\Desktop>java ThrowPro
Welcome to the EVM
Enter your age
14
Exception in thread "main" java.lang.ArithmeticException: You are not allowed to
 vote
        at ThrowPro.main(ThrowPro.java:10)

C:\Users\saif\Desktop>
```

Exception Propagation:

As we all know that the Propagation means transmit one not next.

In the similar when an exception is occurred in a program then it propagates from one method to another method until it is handled or caught.

For example let an exception occur in a program and it is first thrown from the top of the stack if it is not handled. In the next stack stack if exception is not handle there then it will further drop down to the previous method. The process will continue until they reach the very bottom of the call of the stack or it is handled somewhere in the middle.

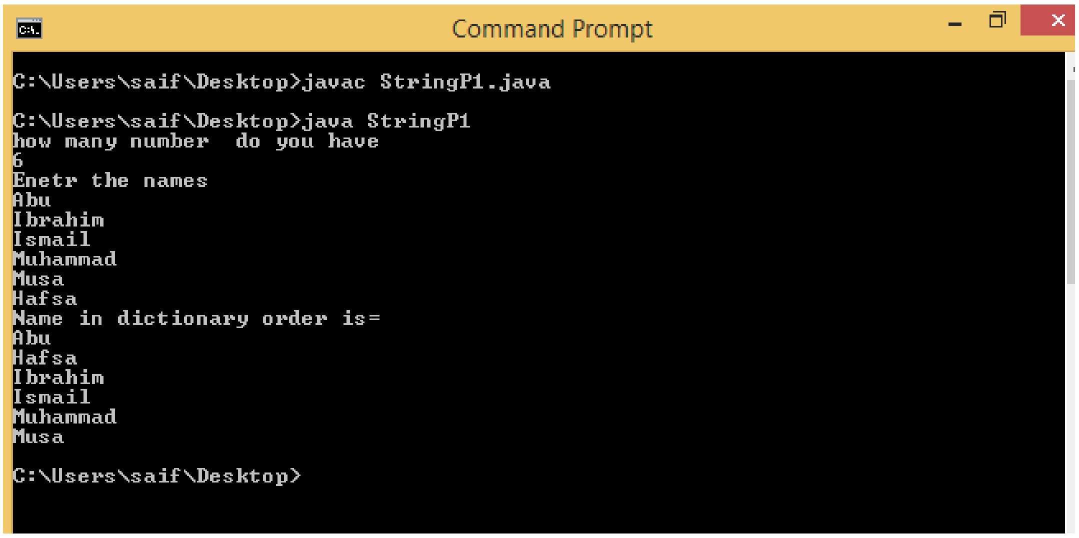
Java Program to print the number in dictionary format:

```
import java.util.*;

class StringP1

{

public static void main(String[] args)
```

```
  {
   Scanner s1=new Scanner(System.in);
   System.out.println("how many number  do you have");
   int n=s1.nextInt();
   String name[]=new String[n];
   Scanner s2=new Scanner(System.in);
   System.out.println("Enetr the names");
   for(int i=0;i<n;i++)
    {
     name[i]=s2.next();
    }
for(int i=0;i<n;i++)
    {
     for(int j=i+1;j<n;j++)
       {
        if(name[i].compareTo(name[j])>0)
         {
          String temp=name[j];
          name[j]=name[i];
          name[i]=temp;
         }
       }
     }
System.out.println("Name in dictionary order is=");
for(int i=0;i<n;i++)
{
System.out.println(name[i]);
}
```

}}

Output:



```
C:\Users\saif\Desktop>javac StringP1.java

C:\Users\saif\Desktop>java StringP1
how many number  do you have
6
Enetr the names
Abu
Ibrahim
Ismail
Muhammad
Musa
Hafsa
Name in dictionary order is=
Abu
Hafsa
Ibrahim
Ismail
Muhammad
Musa

C:\Users\saif\Desktop>
```

Program in java to know the root of the equation:

```
import java.util.*;
class Quadratic
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Quadratic equation should be in the form");
System.out.println("ax^2+bx+c=0");
System.out.println("Enter the value of a, b , c");
double a=in.nextInt();
double b=in.nextInt();
double c=in.nextInt();
double m1=-b/a;
double m2=c/a;
double m4=b*b-4*a*c;
```

if(m4<0)

{System.out.println("Roots are imaginary");}

else

{

double m3=Math.sqrt(m1*m1-4*m2);

double z=m1+m3;

double a1=z/2;

double b1=m1-a1;

System.out.println("The root of following equation " +a+"x^2+"+b+"x+"+c+"=0 is");

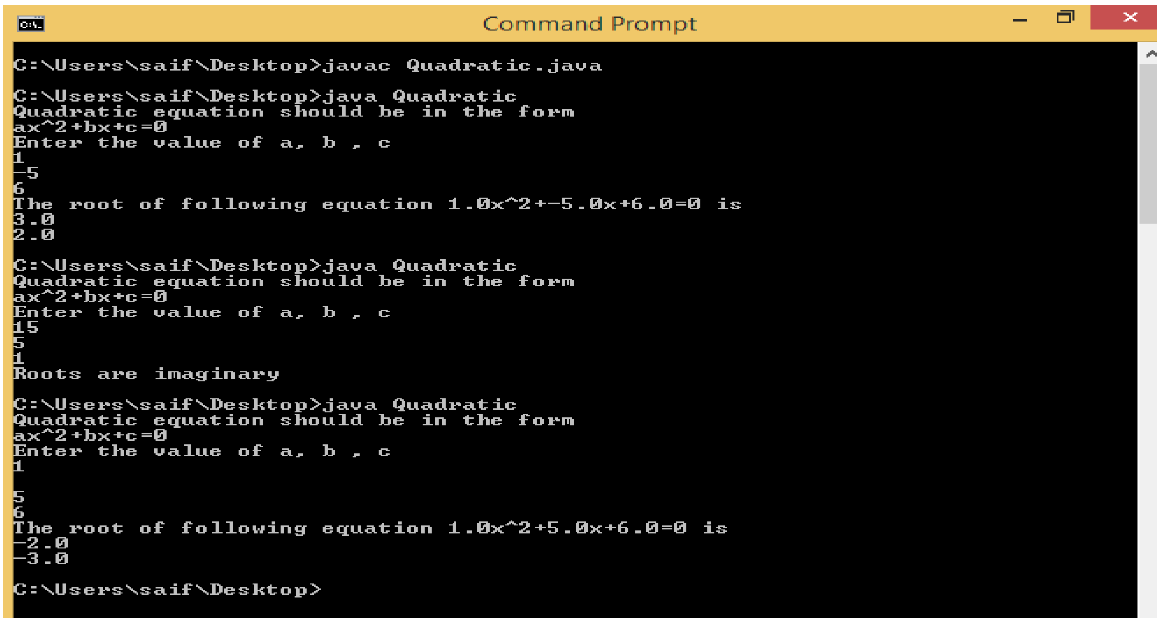
System.out.println(a1);

System.out.println(b1);}

}

}

Output



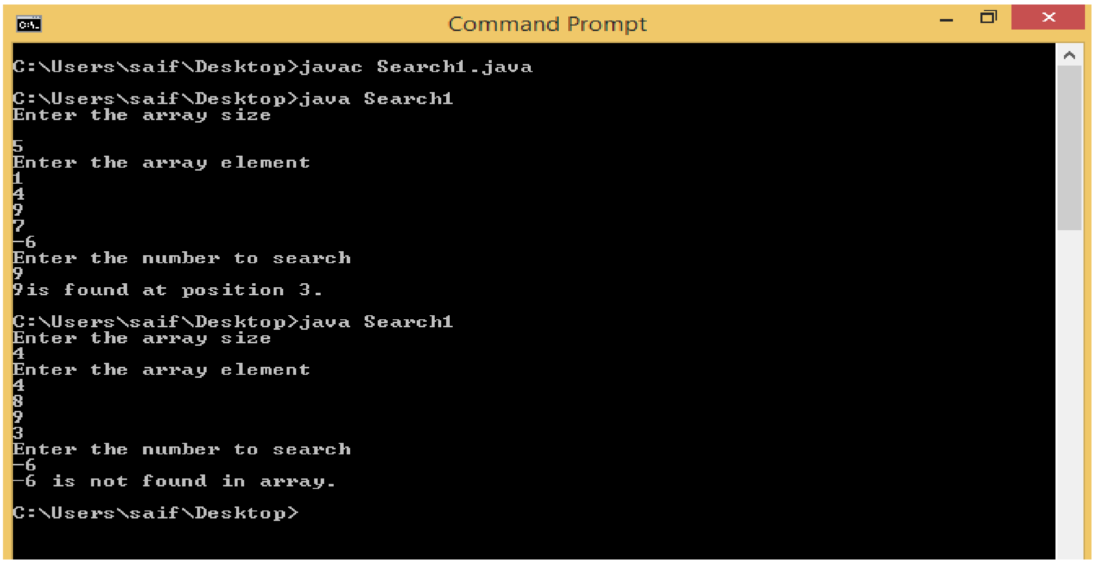
```
C:\Users\saif\Desktop>javac Quadratic.java

C:\Users\saif\Desktop>java Quadratic
Quadratic equation should be in the form
ax^2+bx+c=0
Enter the value of a, b , c
1
-5
6
The root of following equation 1.0x^2+-5.0x+6.0=0 is
3.0
2.0

C:\Users\saif\Desktop>java Quadratic
Quadratic equation should be in the form
ax^2+bx+c=0
Enter the value of a, b , c
15
5
1
Roots are imaginary

C:\Users\saif\Desktop>java Quadratic
Quadratic equation should be in the form
ax^2+bx+c=0
Enter the value of a, b , c
1

5
6
The root of following equation 1.0x^2+5.0x+6.0=0 is
-2.0
-3.0

C:\Users\saif\Desktop>
```

Program to search an element in array

//search an element in java

```
import java.util.*;
class Search1
{
public static void main(String[] args)
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the array size");
int n=in.nextInt();
int a[]=new int[n];
int flag=0;
int p=0;
System.out.println("Enter the array element");
for(int i=0;i<n;i++)
   {
   a[i]=in.nextInt();
   }
System.out.println("Enter the number to search");
int m=in.nextInt();
for(int i=0;i<n;i++)
   {
     if(a[i]==m)
      {
       p=i+1;
      flag++;
      break;
      }
    }
if(flag>0)
{
System.out.println(m+"is found at position "+p+".");
}
else
{
System.out.println(m+" is not found in array.");
}
}
}
```

Output



```
C:\Users\saif\Desktop>javac Search1.java

C:\Users\saif\Desktop>java Search1
Enter the array size

5
Enter the array element
1
4
9
7
-6
Enter the number to search
9
9is found at position 3.

C:\Users\saif\Desktop>java Search1
Enter the array size
4
Enter the array element
4
8
9
3
Enter the number to search
-6
-6 is not found in array.

C:\Users\saif\Desktop>
```

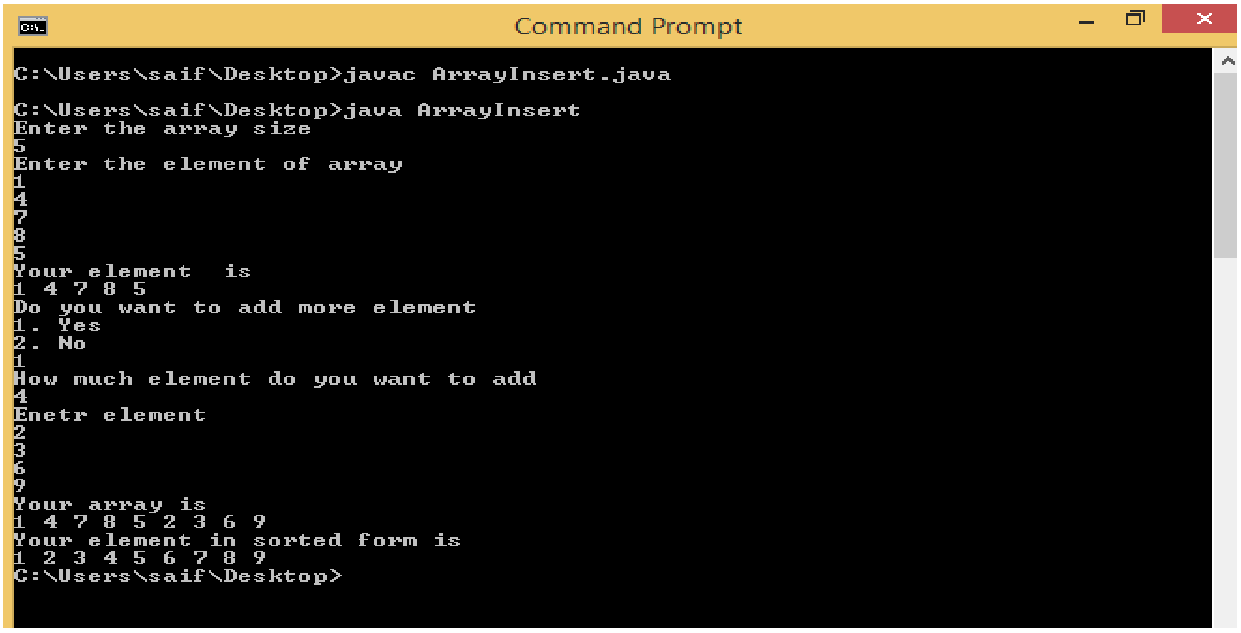
Program to merge two array

```
import java.util.*;
class ArrayInsert
{
public static void main(String[] args)
{
int n=0,m=0;
int a[]=new int[10];
int b[]=new int[10];

int c[]=new int[20];
Scanner in=new Scanner(System.in);
System.out.println("Enter the array size");
n=in.nextInt();
System.out.println("Enter the element of array");
for(int i=0;i<n;i++)
  {
   a[i]=in.nextInt();
   }
System.out.println("Your element in is");
   for(int i=0;i<n;i++)
      {
       System.out .print(a[i]+" ");
       System.out.println("");
       }
System.out.println("Do you want to add more element");
System.out.println("1. Yes");
System.out.println("2. No");
int d=in.nextInt();
if(d==1)
   {
     System.out.println("How much element do you want to add");
     m=in.nextInt();
     System.out.println("Enetr element");
     for(int i=0;i<m;i++)
```

```
        {
         b[i]=in.nextInt();
        }
    }
else
  {
  System.out.println("Thankyou");
  }

int flag=0;
for(int i=0;i<n;i++)
   {
    c[i]=a[i];
    flag++;
   }
    for(int i=0;i<m;i++)
    {
      c[flag++]=b[i];
     }
     int k=m+n;
     System.out.println("Your array is");
     for(int i=0;i<k;i++)
       {
        System.out.print(c[i]+" ");
        }
     for(int i=0;i<k;i++)
        {
         for(int j=i+1;j<k;j++)
```

```
            {
              if(c[i]>c[j])
               {
                int temp=0;
                temp=c[j];
                c[j]=c[i];
                c[i]=temp;
               }
             }
         }
  System.out.println("Your element in sorted form is");
     for(int i=0;i<k;i++)
        {
        System.out.println(c[i]+" ");
        }
}
}
```

Output:



```
C:\Users\saif\Desktop>javac ArrayInsert.java

C:\Users\saif\Desktop>java ArrayInsert
Enter the array size
5
Enter the element of array
1
4
7
8
5
Your element  is
1 4 7 8 5
Do you want to add more element
1. Yes
2. No
1
How much element do you want to add
4
Enetr element
2
3
6
9
Your array is
1 4 7 8 5 2 3 6 9
Your element in sorted form is
1 2 3 4 5 6 7 8 9
C:\Users\saif\Desktop>
```

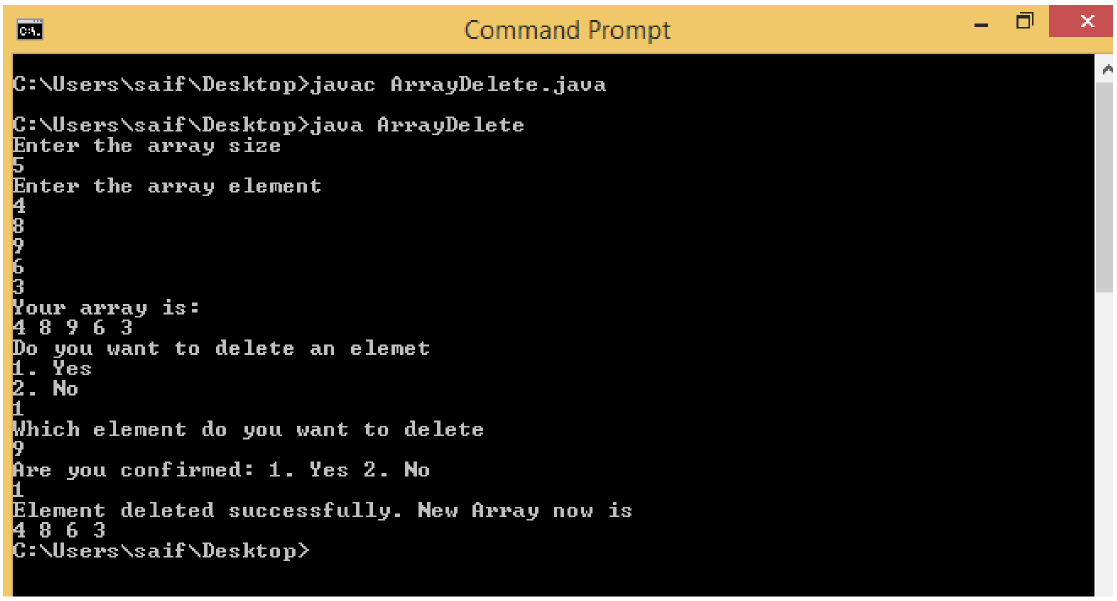
Program to delete an element from array:

```
import java.util.*;
class ArrayDelete
{
public static void main(String[] args)
{
int n,c,y;
int d=0;
int flag=0;
int a[]=new int[10];
Scanner in=new Scanner(System.in);
System.out.println("Enter the array size");
n=in.nextInt();
System.out.println("Enter the array element");
for(int i=0;i<n;i++)
{
a[i]=in.nextInt();
}
```

```
System.out.println("Your array is:");
for(int i=0;i<n;i++)
{
System.out.print(a[i]+" ");
}
System.out.println("");
System.out.println("Do you want to delete an elemet");
System.out.println("1. Yes");
System.out.println("2. No");
c=in.nextInt();
  if(c==1)
          {
             System.out.println("Which element do you want to delete");
             d=in.nextInt();
             for(int i=0;i<n;i++)
                              {
                               if(a[i]==d)
                                         {
                                          flag++;
                                         }
                                  }
          }
   else if(c==2)
        {
         System.out.println("ThankYou");
         System.exit(0);
         }
   else
```

```
        {
         System.out.println("choose 1 or 2");
        }
if(flag>0)
{
System.out.println("Are you confirmed: 1. Yes 2. No");
                                        y=in.nextInt();
                                            if(y==1)
                                                 {
                                                   System.out.println("Element deleted successfully. New Array
now is");
                                                        for(int j=0;j<n;j++)
                                                                 {
                                                                   if(a[j]!=d)
                                                                         {
                                                                          System.out.print(a[j]+" ");
                                                                          }
                                                                    else
                                                                          {
                                                                           System.out.print("");
                                                                           }
                                                                   }
                                                   }
                                                  else
                                                   {
                                                     for(int k=0;k<n;k++)
                                                                 {
                                                                  System.out.print("Your array is");
                                                                  System.out.print(a[k]+" ");
                                        }
                               }

}
else
{
System.out.println("Elements are not found. Please try other.");
}
}
}
```

Output:



```
C:\Users\saif\Desktop>javac ArrayDelete.java

C:\Users\saif\Desktop>java ArrayDelete
Enter the array size
5
Enter the array element
4
8
9
6
3
Your array is:
4 8 9 6 3
Do you want to delete an elemet
1. Yes
2. No
1
Which element do you want to delete
9
Are you confirmed: 1. Yes 2. No
1
Element deleted successfully. New Array now is
4 8 6 3
C:\Users\saif\Desktop>
```

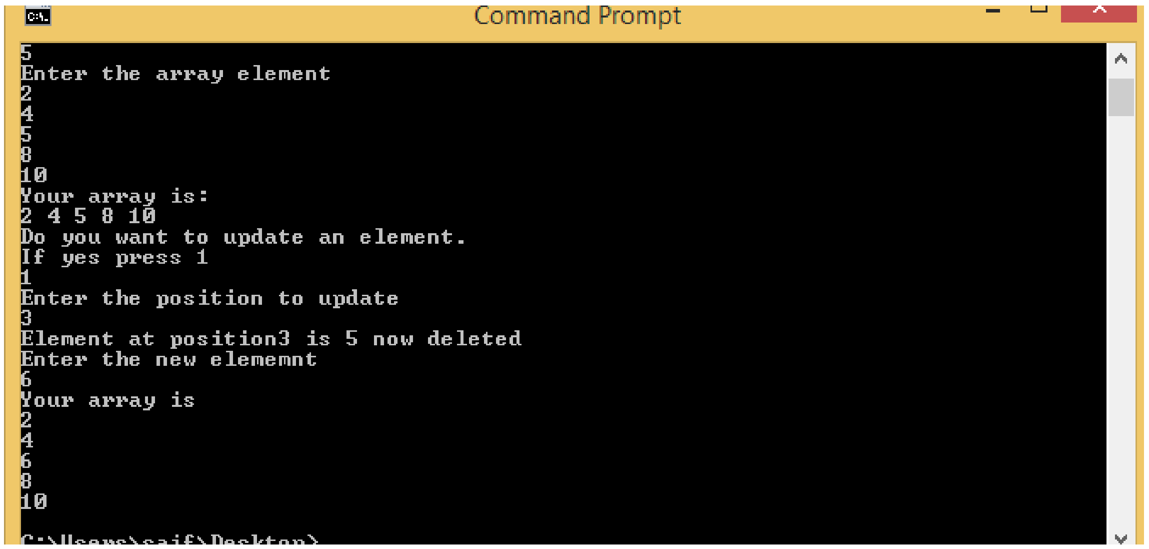
Program 2 to update an element:

```
import java.util.*;
class ArrayUpdate
{
public static void main(String[] args)
```

```
{
Scanner in=new Scanner(System.in);
System.out.println("Enter the array size");
int n=in.nextInt();
int a[]=new int[n];
System.out.println("Enter the array element");
for(int i=0;i<n;i++)
{
a[i]=in.nextInt();
}
System.out.println("Your array is:");
for(int i=0;i<n;i++)
{
System.out.print(a[i]+" ");
}
System.out.println("");
System.out.println("Do you want to update an element.");
System.out.println("If yes press 1");
int p=in.nextInt();
if(p==1)
    {
        System.out.println("Enter the position to update");
        int po=in.nextInt();
        int x=0;
        x=po-1;
        if(po<n)
            {
            System.out.println("Element at position"+po+" is "+a[x]+" now deleted");
```

```
                System.out.println("Enter the new elememnt");
                int nw=in.nextInt();
                a[x]=nw;

               }
         else
               {
                System.out.println("out of the array position");
               }
          }
else
      {
       System.out.println("ThankYou");
      }
System.out.println("Your array is");
for(int i=0;i<n;i++)
{
System.out.println(a[i]+" ");
}

}}
```

Output:



```
5
Enter the array element
2
4
5
8
10
Your array is:
2 4 5 8 10
Do you want to update an element.
If yes press 1
1
Enter the position to update
3
Element at position3 is 5 now deleted
Enter the new elememnt
6
Your array is
2
4
6
8
10

C:\Users\saif\Desktop>
```

String:

A string is a sequence of character which is used to store the array of character in a variable but in java a string is a object which is also usd to store the character in sequence. Java provide the String class to perform the operation on the string e.g. compare the string , find the ;ength of the string , concatenade the string and so on.

1. Java String is present in the java.lang package
2. String pool will made only for the literal type string.
3. In liertal case declaration object will always create inside the pool and for the same content they will refer to same memory but in case of the object creation declaration the object memory will reside inside the memory.

In Java a string can bbe define in the two different way

1. String Literal
2. String object

String literal:

To effiviently manage the memory in a program we use the string literal method to decalare a string.In string literal method string can be declare as the

String s1="I am Java";

String s2="I am Java";

String s3="I am Python";

When we declare the string literal in the java program then the jvm check the content in the string pool. If the content is same then no seprate memory is allocated but a referance is created of the object having the same content.

Since s1 and s2 have the same content "I am Java " so they both point to the same memory allocation bur remember if we change the content of one object then it will also effect the content of the another object having the same content before and share same memory.Thos make the string final in java.

String Object

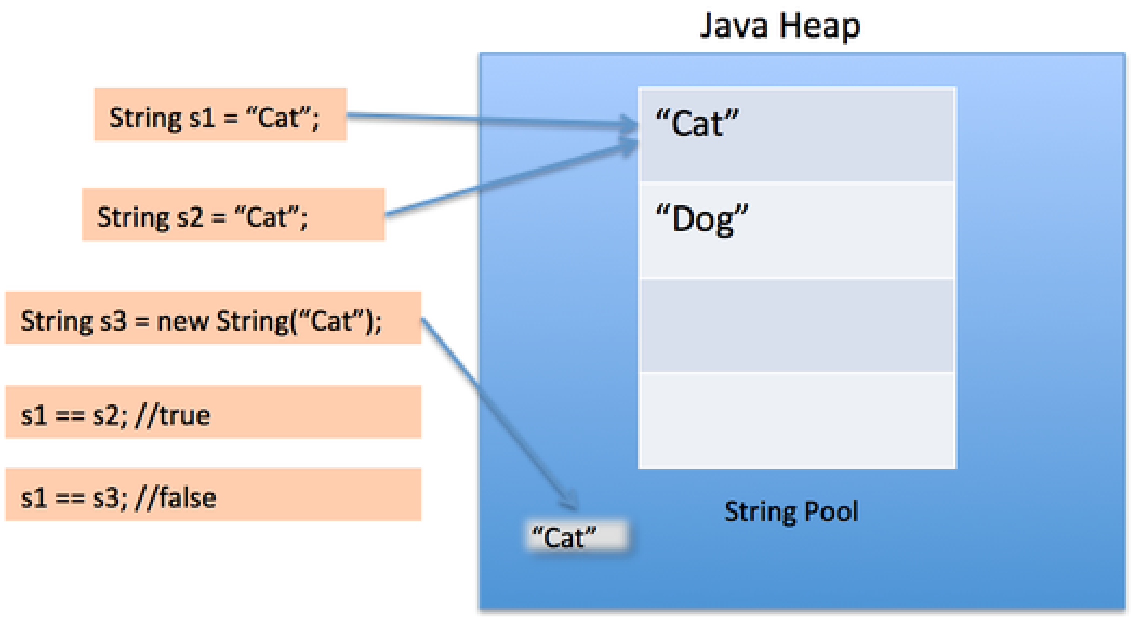
String object is another type of the declaring staring using the new object.

String s1=new String("I am Java");

String s1=new String("I am java");

String s1=new String("I Am PythInon");

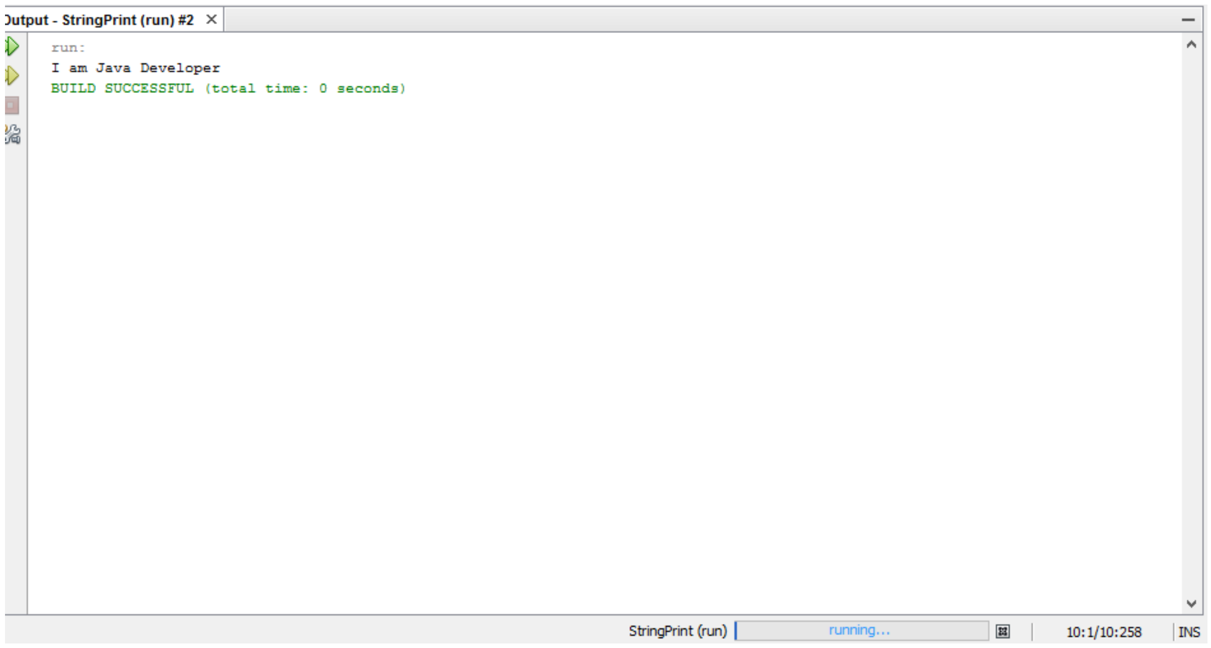
In this case separte memory will allcate to the separete object and no amtter theny have same or different content in the object.



Program: How to define and print a string in Java

```
import java.util.*;
public class StringPrint {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
       String p1="I am Java";
       String p3=new String("Developer");
       System.out.println(p1+" "+p3);
       }
}
```

Output:



Output - StringPrint (run) #2

```
run:
I am Java Developer
BUILD SUCCESSFUL (total time: 0 seconds)
```

StringPrint (run) running... 10:1/10:258 INS

Compare the string in Java

We can compare the string in three ways as given below:

1. By equals() method
2. By == operator
3. And by compareTo method.

By equals() method we compare the content of the string. Generally equlas() is for the value of reference to compare but not for the content so we override the method to compare the value of content.

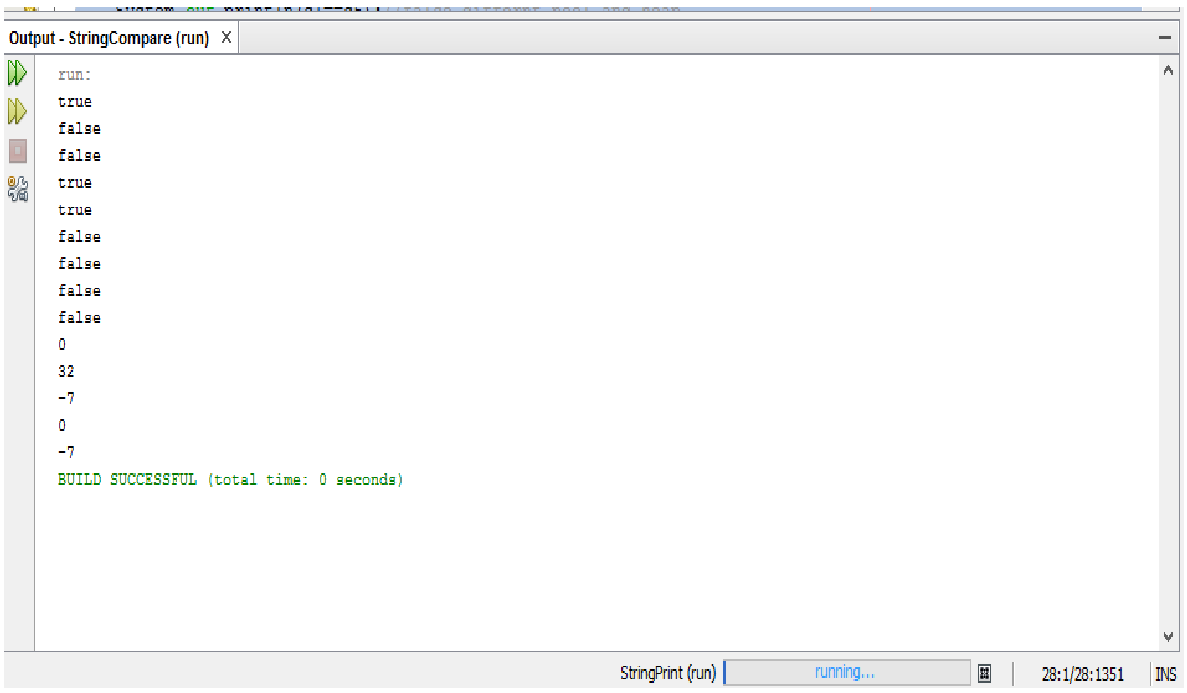
The == is used for the compare whether the string are from the same string pool or from the same heap memory along with the string content. If two string is from the same string pool or heap then the resultant will be the true else the resultant will be the false.

compareTo() function is used for the check the acceneding order or alphabetical order of the string.

Program: Java program to demonstrate the equlas(), == and compareTo() method in java for string comparison

```
import java.util.*;

public class StringCompare {

   public static void main(String[] args) {

   String s1="India";

   String s2="India";

   String s3="INDIA";

   String s4="Pakistan";

   String s5=new String("India");

   String s6=new String("India");

   String s7=new String("INDIA");

   String s8=new String("Pakistan");

   System.out.println(s1.equals(s2));//result will true India in India is same

   System.out.println(s1.equals(s3));//false result as india(small) and INDIA(capital)

   System.out.println(s1.equals(s4));//false India & Pakistan not same

   System.out.println(s1.equals(s5));//true India & india are same

   System.out.println(s1==s2);// true for same pool

   System.out.println(s1==s3);//flase because different string at same pool

   System.out.println(s1==s5);//false differnt pool and heap

   System.out.println(s3==s7);//false diffrent values and pool-heap
   System.out.println(s1==s4);//false different value and pool-heap
   System.out.println(s1.compareTo(s2));// 0 for same string
   System.out.println(s1.compareTo(s3));// 32 differ i & I ascii value
   System.out.println(s1.compareTo(s4));//-7 Because I less then P by -7 in ascii value
   System.out.println(s1.compareTo(s5));// same india in India in pool & heap
   System.out.println(s1.compareTo(s8));//-7 differ i & P ascii value
}
}
```

Output



```
run:
true
false
false
true
true
false
false
false
false
0
32
-7
0
-7
BUILD SUCCESSFUL (total time: 0 seconds)
```

Java String Concatenation:

Java cancatenation is done by the concat(). With the help of this method we jjoin the two string together in the form of new string.In two string concatenates the second string willl be add at the end of the first string.

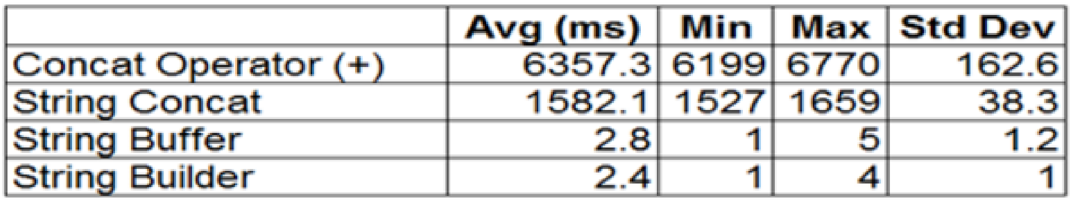
Java string concatenation can be done by the following four way:

1. Concatenation operator (+)
2. StringBuffer class

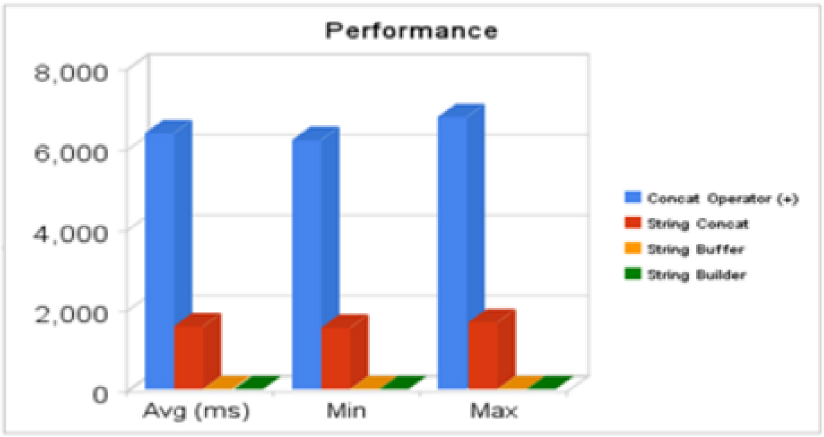
3. StringBuilder class
4. String.concat() function

The best one is used to “+” because it is simple to add string as compare to the other method but in case of the perfomance where we are going to add the thousand charater together then this opeartor will not use because of poort speed and poor memory management. The “+ ” opearator can aslo be never used in the loop.

The perfomance of the string cancatenation is given below by the all four methood



	Avg (ms)	Min	Max	Std Dev
Concat Operator (+)	6357.3	6199	6770	162.6
String Concat	1582.1	1527	1659	38.3
String Buffer	2.8	1	5	1.2
String Builder	2.4	1	4	1



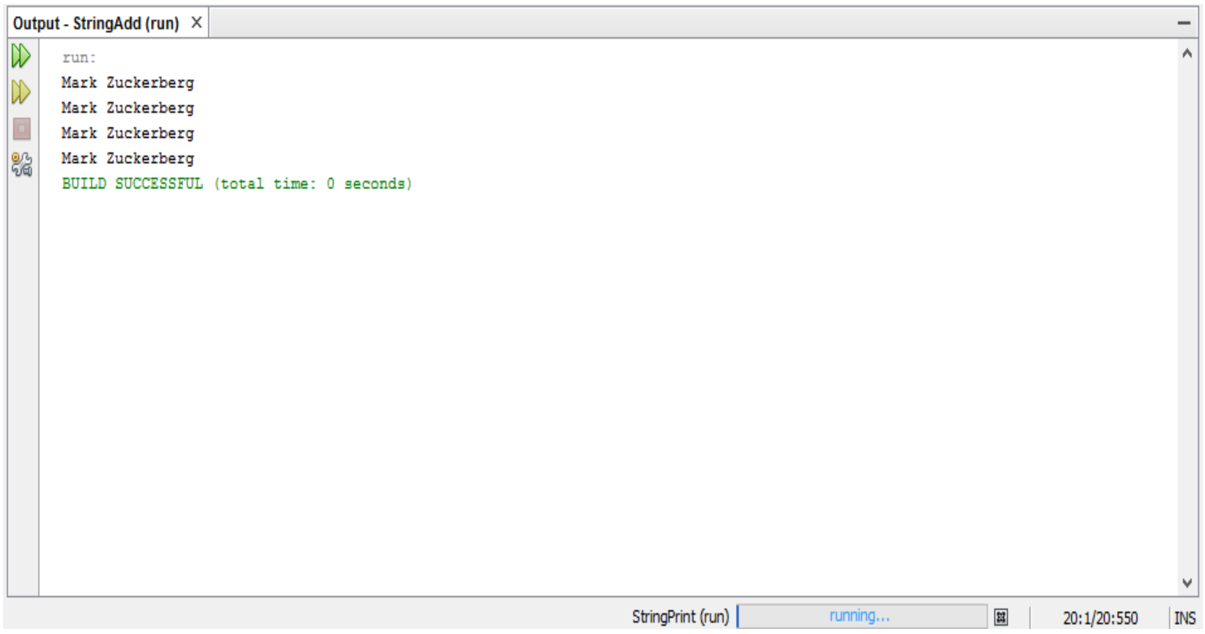
So to add multiple string always go for the String Builder in java to add string fast.

Program to add concatenation of two string by four method:

```
import java.util.*;
public class StringAdd {
public static void main(String[] args) {
    String s1="Mark";
    String s2="Zuckerberg";
    String s3=s1+" "+s2;
```

```
        System.out.println(s3);
        String s4=s1.concat(" ").concat(s2);
        System.out.println(s4);
        StringBuffer s5=new StringBuffer();
        s5.append(s1).append(" ").append(s2);
        System.out.println(s5);
        StringBuilder s6=new StringBuilder();
        s6.append(s1).append(" ").append(s2);
        System.out.println(s6);
            }
}
```

Output:



Output - StringAdd (run)

```
run:
Mark Zuckerberg
Mark Zuckerberg
Mark Zuckerberg
Mark Zuckerberg
BUILD SUCCESSFUL (total time: 0 seconds)
```

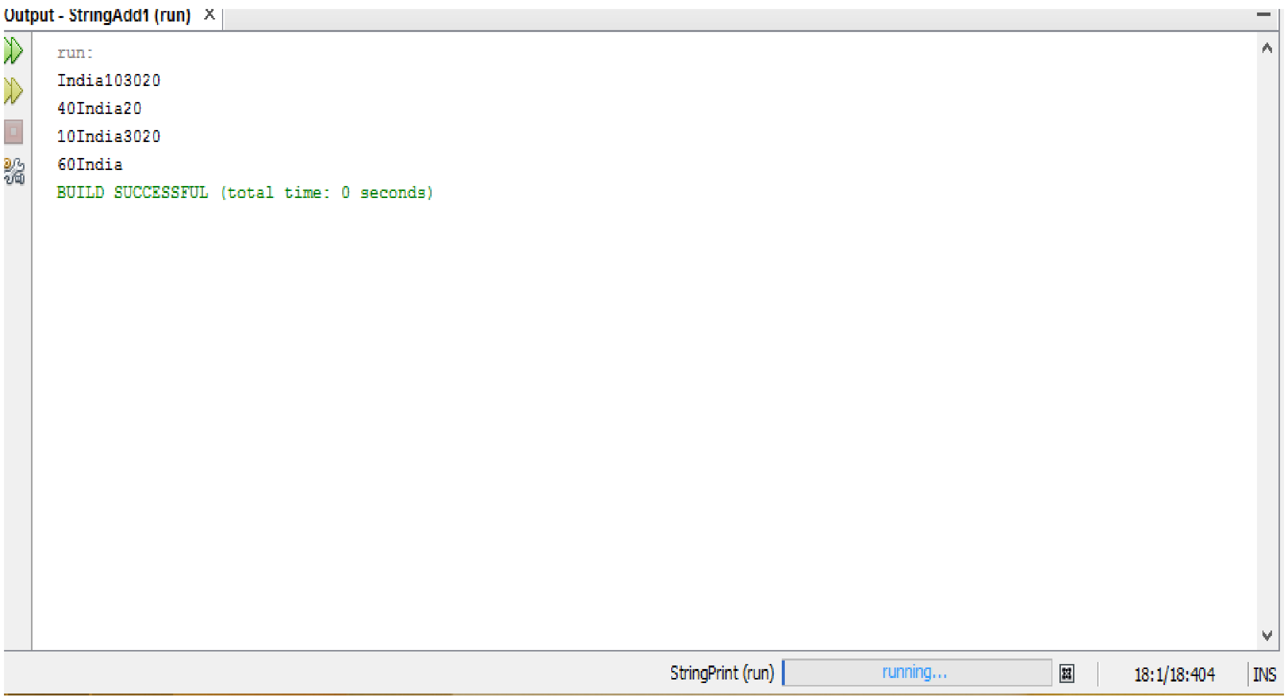
StringPrint (run) running... 20:1/20:550 INS

Program to Understand the + operator in Java concatenation

```
import java.util.*;
public class StringAdd1 {
    public static void main(String[] args) {
```

```
    String s="India";
    int a=10;
    int b=30;
    int c=20;
    String s1=s+a+b+c;
    String s2=a+b+s+c;
    String s3=a+s+b+c;
    String s4=a+b+c+s;
    System.out.println(s1);
    System.out.println(s2);
    System.out.println(s3);
    System.out.println(s4);
  }
}
```

Output:



```
run:
India103020
40India20
10India3020
60India
BUILD SUCCESSFUL (total time: 0 seconds)
```

Application of concatenation:

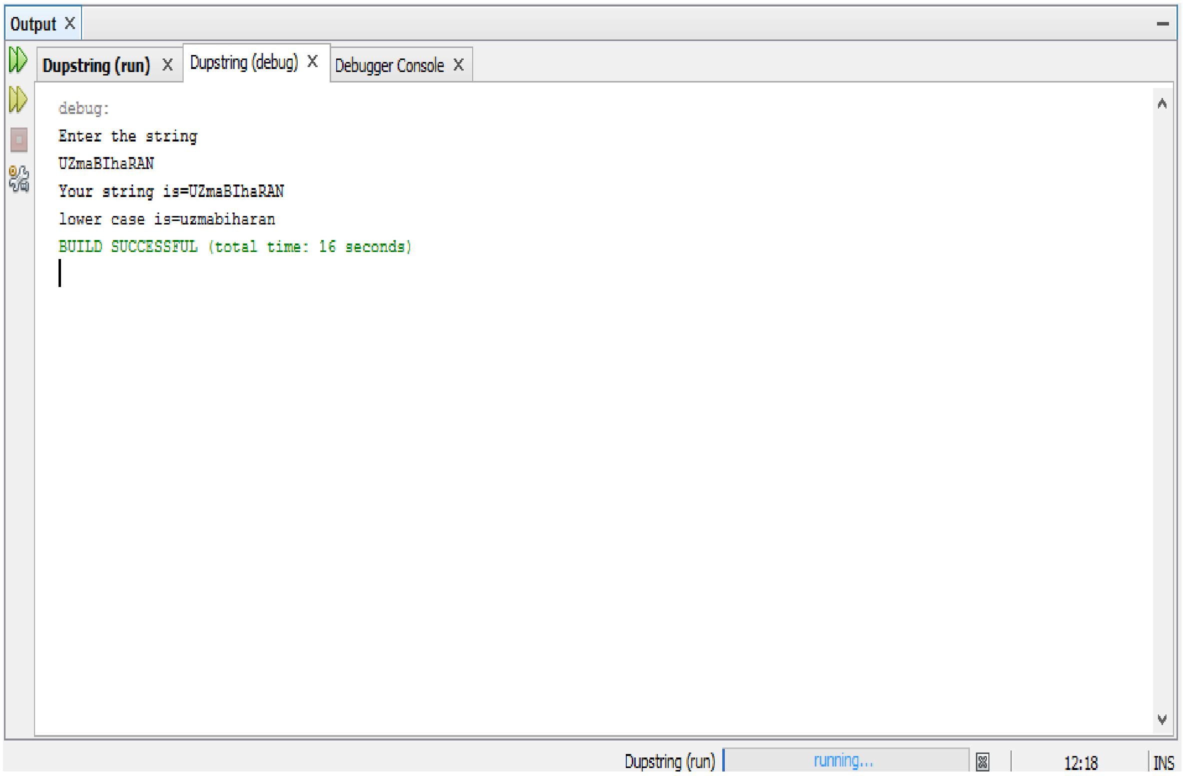
1. It help in the program to display the name of the person by concatenate his first and the last name.
2. It help to arrange the name in the dictionary order.

Program to convert UpperCase String into lowerCase string

```
import java.util.*;
public class Dupstring {

    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        String str;
        System.out.println("Enter the string");
      str=in.next();
      System.out.println("Your string is="+str);
      char ch[]=new char[str.length()];
       for(int i=0;i<str.length();i++)
       {
       ch[i]=str.charAt(i);
       }
       for(int i=0;i<str.length();i++)
       {
       if(ch[i]>=65 && ch[i]<=90)
       {
       ch[i]=(char)((ch[i]+32));
       }
       }
       System.out.print("lower case is=");
       for(int i=0;i<str.length();i++)
       {
       System.out.print((char)+ch[i]);
```

```
            }
     System.out.println();
     }

}
```

OutPut:



```
debug:
Enter the string
UZmaBIhaRAN
Your string is=UZmaBIhaRAN
lower case is=uzmabiharan
BUILD SUCCESSFUL (total time: 16 seconds)
```

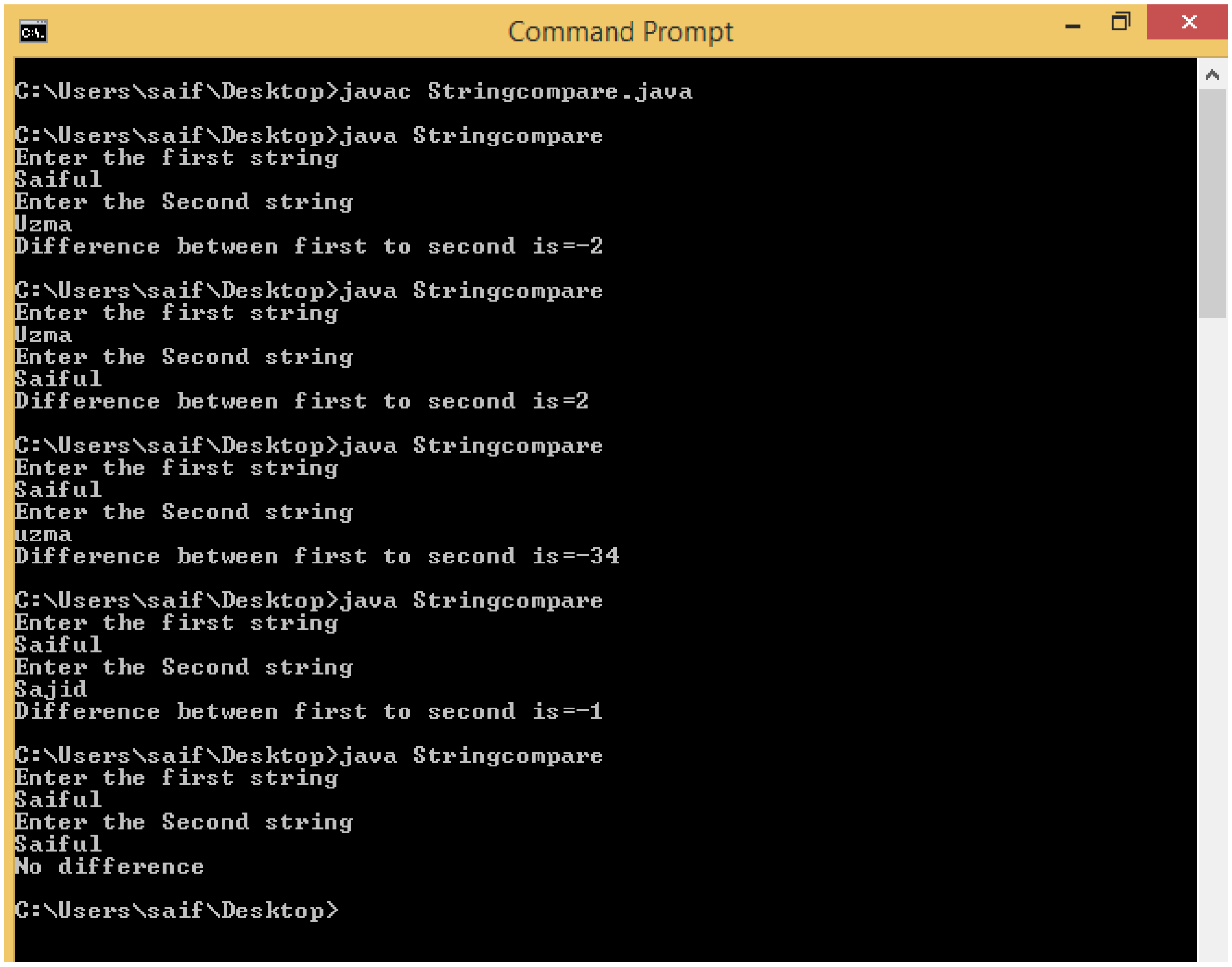
Program for Comparison between the String:

```
import java.util.*;
public class Stringompare {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the first string");
        String a=in.next();
        System.out.println("Enter the first string");
        String b=in.next();
        int l1=a.length();
        int l2=b.length();
        char achar[]=new char[l1];
        char bchar[]=new char[l2];
        for(int i=0;i<l1;i++)
        {
        achar[i]=a.charAt(i);
        }
        for(int i=0;i<l2;i++)
        {
        bchar[i]=b.charAt(i);
        }
        int l;

        if(l1>l2)
        {
        l=l2;
        }
        else
        {
        l=l1;
        }
          int r[]=new int[l];
        int flag=l;
        for(int i=0;i<l;i++)
        {
```

```
        if(achar[i]==bchar[i])
        {
        flag--;
        }
        else if(achar[i]>bchar[i])
        {
        r[i]=(achar[i]-bchar[i]);
        }
        else
        {
        r[i]=bchar[i]-achar[i];
        }
        }
        if(flag==0)
           {
            System.out.println("No difference");
           }
        else
        for(int i=0;i<l;i++)
        { if(r[i]!=0)
        {
        System.out.println(r[i]);
        break;
       }



        }
```

```
  }}
```

Output:



```
C:\Users\saif\Desktop>javac Stringcompare.java

C:\Users\saif\Desktop>java Stringcompare
Enter the first string
Saiful
Enter the Second string
Uzma
Difference between first to second is=-2

C:\Users\saif\Desktop>java Stringcompare
Enter the first string
Uzma
Enter the Second string
Saiful
Difference between first to second is=2

C:\Users\saif\Desktop>java Stringcompare
Enter the first string
Saiful
Enter the Second string
uzma
Difference between first to second is=-34

C:\Users\saif\Desktop>java Stringcompare
Enter the first string
Saiful
Enter the Second string
Sajid
Difference between first to second is=-1

C:\Users\saif\Desktop>java Stringcompare
Enter the first string
Saiful
Enter the Second string
Saiful
No difference

C:\Users\saif\Desktop>
```

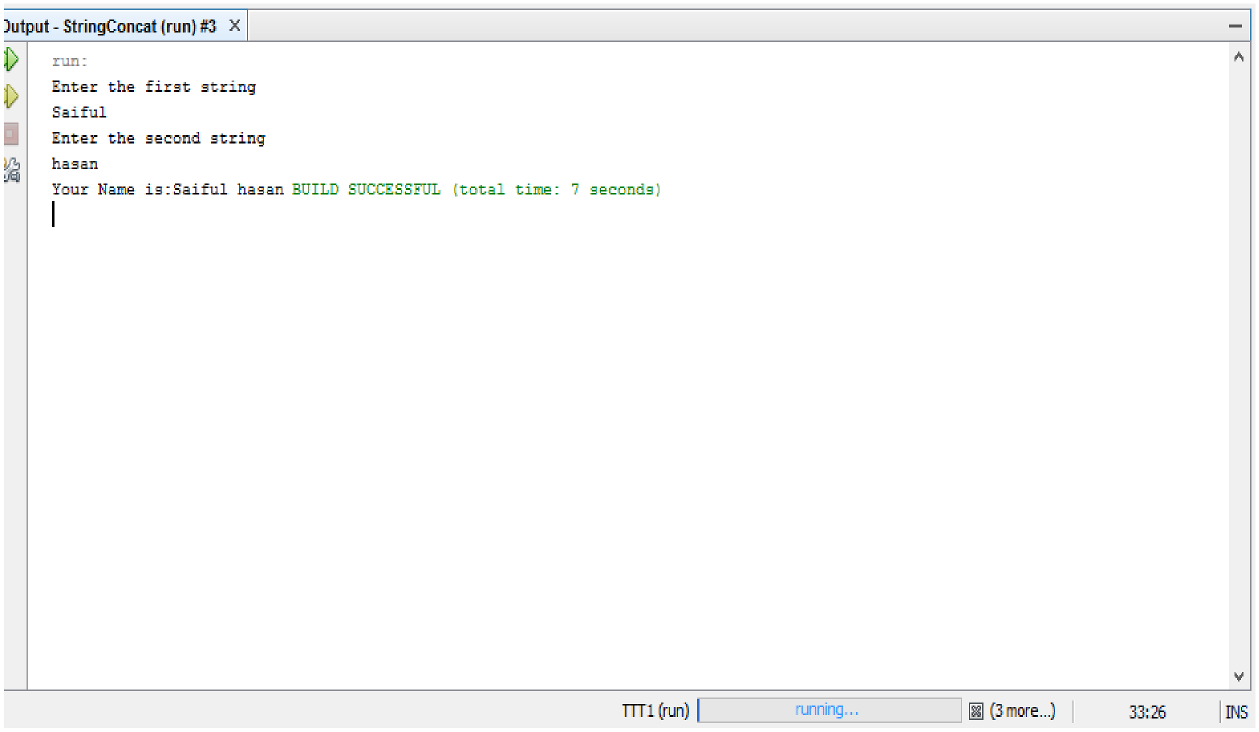
Program to add two string without using concat methid()

```
import java.util.*;
public class StringConcat {
  public static void main(String[] args) {
    Scanner in=new Scanner(System.in);
    System.out.println("Enter the first name");
```

```
String fname=in.next();
char name1[]=new char[fname.length()];
System.out.println("Enter the second name");
String lname=in.next();
char name2[]=new char[lname.length()];
for(int i=0;i<fname.length();i++)
{
name1[i]=fname.charAt(i);
}
for(int i=0;i<lname.length();i++)
{
name2[i]=lname.charAt(i);
}
char name[]=new char[30];
int flag=0;
for(int i=0;i<fname.length();i++)
        {
        name[i]=name1[i];
        flag++;
        }
flag=flag+1;
 for(int i=0;i<lname.length();i++)
        {
        name[flag]=name2[i];
        flag++;
        }
for(int i=0;i<=flag;i++)
```

```
        {
         if(i!=fname.length())
           {
            System.out.print(name[i]);
           }
        else
           {
            System.out.print(" ");
           }
        }
        }
    }
```

Output:



```
run:
Enter the first string
Saiful
Enter the second string
hasan
Your Name is:Saiful hasan BUILD SUCCESSFUL (total time: 7 seconds)
```

Program:

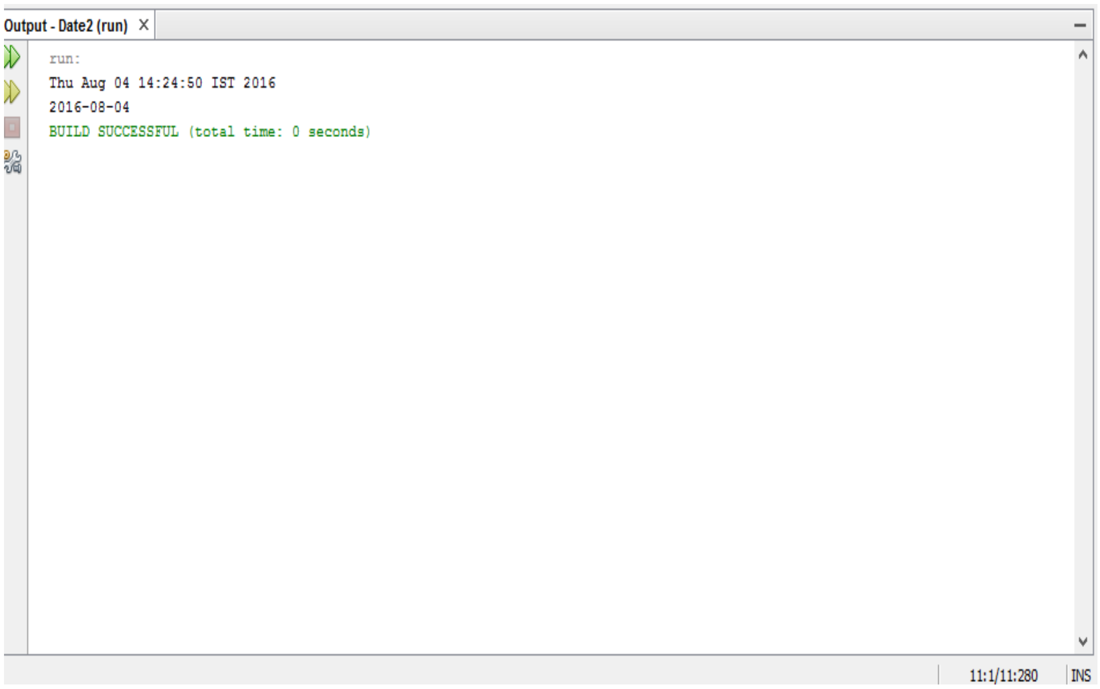
Java Date:

To represent the date time in the enterprice application we need to learn about date and time. Java provode the date and time API so we do need need to write the code for it. The package hjava.util.sql is used to represent the date while on the other hand the package java.util.Date help us to reprent date along with the time.This is the difference between these two classes. The java.sql.Date inherit the java.util.Date class it is widely used in jdbc because it represent the date that can be stored in the data base.

Program for Date in Java:

```
public class Date2 {
    public static void main(String[] args) {
      java.util.Date date1=new java.util.Date();
      long m=System.currentTimeMillis();
      java.sql.Date date2=new java.sql.Date(m);
      System.out.println(date1);
    System.out.println(date2);
    }
   }
```

Output:


Output - Date2 (run)

```
run:
Thu Aug 04 14:24:50 IST 2016
2016-08-04
BUILD SUCCESSFUL (total time: 0 seconds)
```

11:1/11:280 INS

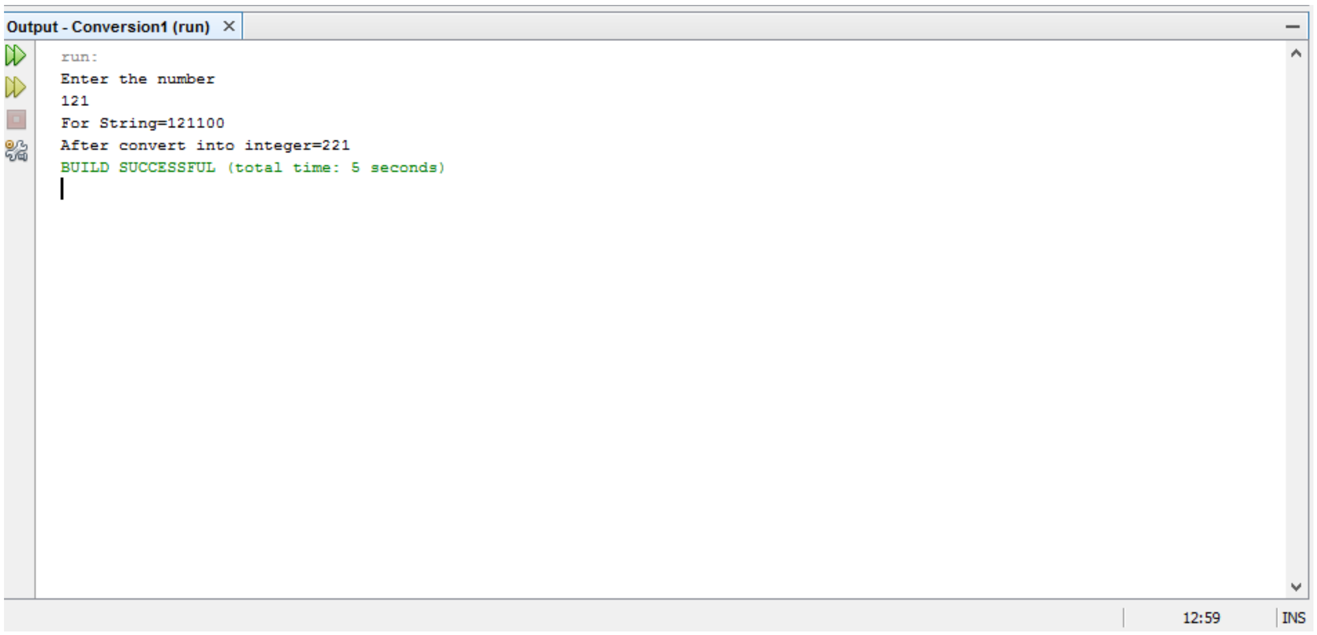
Conversion in Java:

1. String to integer():

Program:

```
import java.util.*;
public class Conversion1 {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the number");
        String a=in.next();
        int a1=Integer.parseInt(a);
        a=a+100;
        a1=a1+100;
        System.out.println("For String="+a);
       System.out.println("After converting into integer"+a1);
    }
}
```

Output:



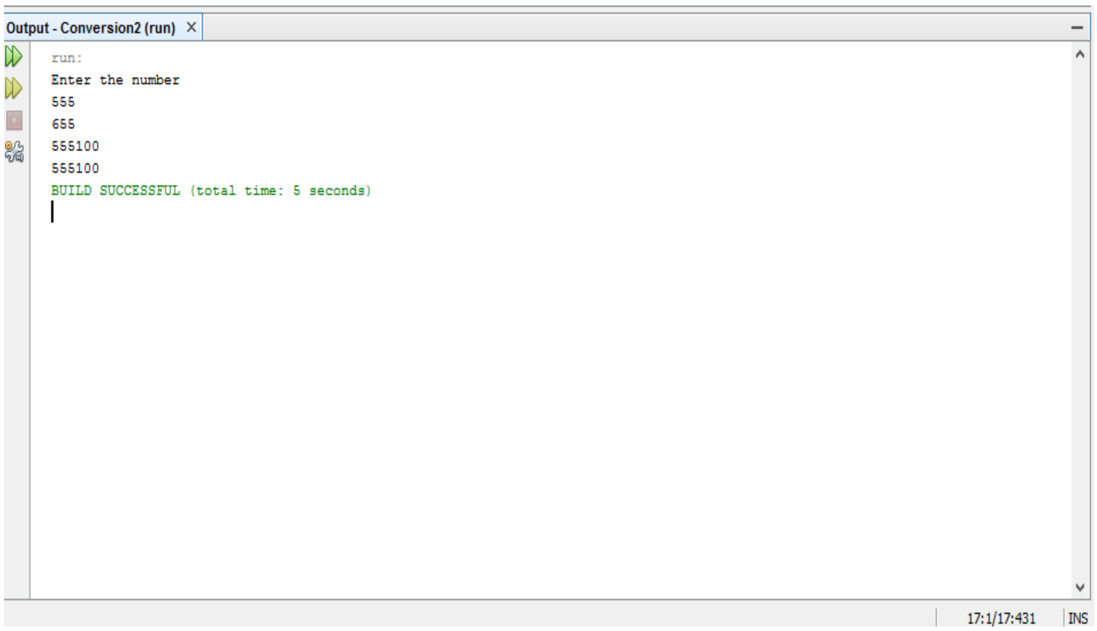
```
run:
Enter the number
121
For String=121100
After convert into integer=221
BUILD SUCCESSFUL (total time: 5 seconds)
```

Conversion: Integer to String

Program:

```
import java.util.*;
public class Conversion2 {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the number");
        int n=in.nextInt();
        String s=String.valueOf(n);
        String s1=Integer.toString(n);
        n=n+100;
        s=s+100;
        s1=s1+100;
        System.out.println(n);
        System.out.println(s);
    System.out.println(s1);
  }
}
```

Output:



```
run:
Enter the number
555
655
555100
555100
BUILD SUCCESSFUL (total time: 5 seconds)
```

Difference between the parseInt and valueOf method in Java:

1. Integer.valueOf() returns an Integer object, while Integer.parseInt() returns an int primitive.
2. Value of use the parseInt method internally.

Program for comparision between the two string

```
import java.util.*;
public class StringEquals {
  public static void main(String[] args) {
    Scanner in=new Scanner(System.in);
    System.out.println("Enter the first string");
    String s1=in.next();
    System.out.println("Enter the second string");
    String s2=in.next();
```

```
char ch1[]=new char[20];
char ch2[]=new char[20];
for(int i=0;i<s1.length();i++)
{
ch1[i]=s1.charAt(i);
}
for(int i=0;i<s2.length();i++)
{
ch2[i]=s2.charAt(i);
}
int l;
if(s1.length()>s2.length())
{
l=s1.length();
}
else
{
l=s2.length();
}
int flag=0;
for(int i=0;i<l;i++)
{
if(ch1[i]!=ch2[i])
{
flag++;
}
}
```

```
    if(flag==0)

    {

    System.out.println("false");

    }

    else

    {

    System.out.println("true");

    }

    }

  }
```

Output:

File Handling in Java:

Java.io

In the early of the 1995 the file handling is the important topic. So when the first version of java 1.0 is released in 1995 by Sun then file Java.io was one of the most important topic in java. The java.io is used to create the file, read the file and write the file and so on. As later after the introduce of database like Oracle and MySql we move to the database to store the data and perform the query and java.io was skip from the syllabus. But we know the oracle and mysql is used to handle the large data so it is not sound good to install the database and purchase the license to store the small data and files. So again java.io is became the important topic for the developer who value the time.

We are going to cover the following topics and operation method related to them in java

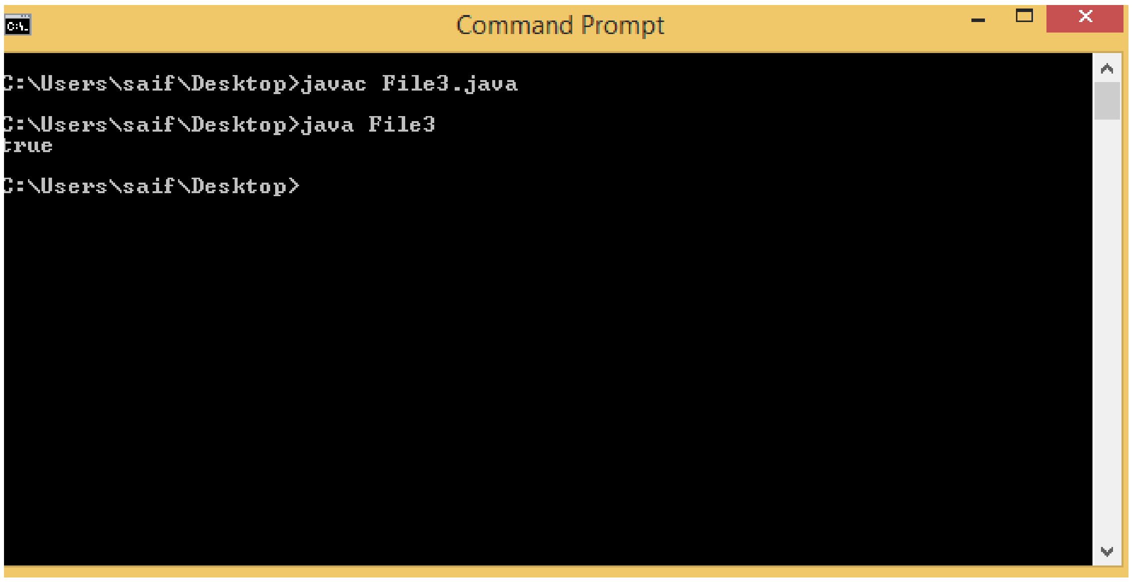
1. File
2. File Writer
3. File Reader
4. Buffered Writer

5. Buffered Reader
6. Print Writer

Program: To create a file into your current directory.

```
import java.io.*;

class File3

{

public static void main(String[] args) throws Exception

{

 File f=new File("robot.txt");

 f.createNewFile();

 System.out.println(f.exists());

}

}
```

Output:



Command Prompt

```
C:\Users\saif\Desktop>javac File3.java

C:\Users\saif\Desktop>java File3
true

C:\Users\saif\Desktop>
```

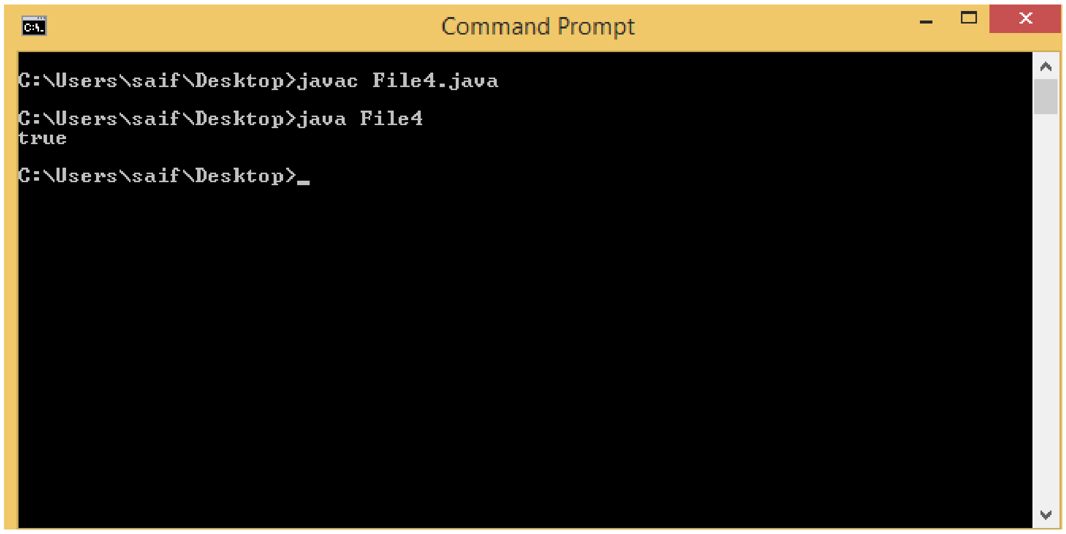
True reprent that the file has been created in the directory with the name robot.txt

In the program the code line File f=new File("robot.txt"); is not responsible to create the file but it check if the file name is exist or not. If a file is exist with the same name ex. Robot.txt then it will not further proceed to create a file but if a file is not present then a new file will be created with name robot.txt in the current directory and this directory will be pointed by the object f..

Program to create a directory into your current directory:

```
import java.io.*;

class File4

{

public static void main(String[] args)

{

File f=new File("Math");

f.mkdir();

System.out.println(f.exists());

}

}
```

Output:



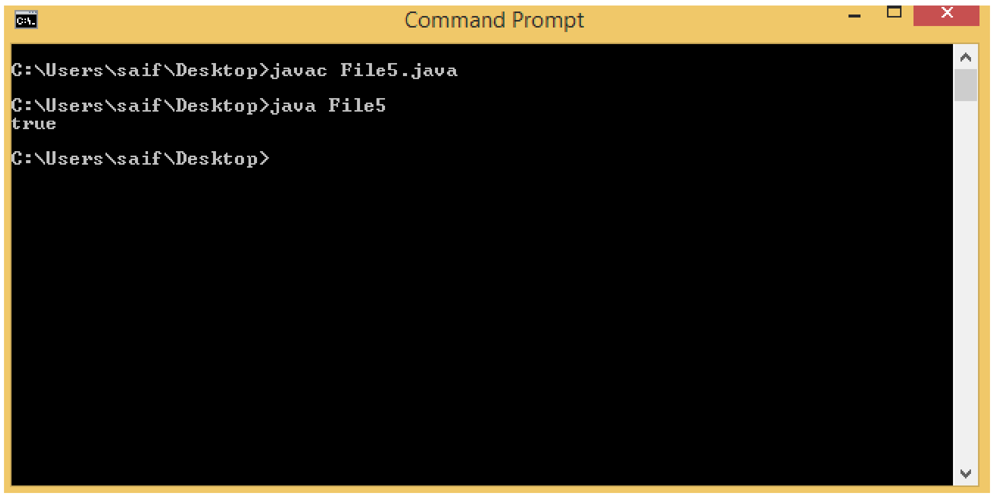
```
C:\Users\saif\Desktop>javac File4.java

C:\Users\saif\Desktop>java File4
true

C:\Users\saif\Desktop>_
```

New Directory math will be created.

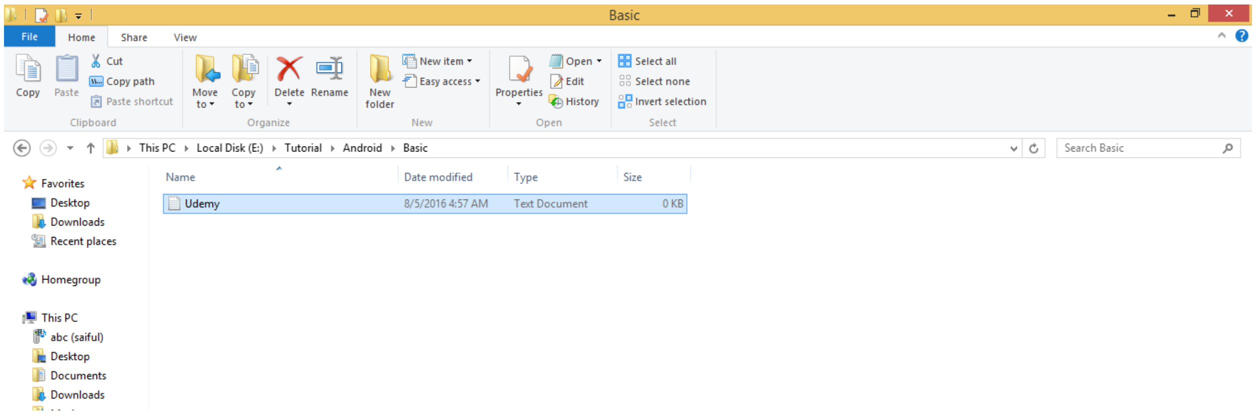
Program to create a file into new Directory:

```
import java.io.*;
class File5
{
public static void main(String[] args) throws Exception
{
File f=new File("E:\\Tutorial\\Android\\Basic","Udemy.txt");
f.createNewFile();
System.out.println(f.exists());
}
}
```

Output:



```
C:\Users\saif\Desktop>javac File5.java

C:\Users\saif\Desktop>java File5
true

C:\Users\saif\Desktop>
```



Name	Date modified	Type	Size
Udemy	8/5/2016 4:57 AM	Text Document	0 KB

Create the file at specified directory.

File Writer:

File Writer is used to write the character or text data into the file. We have following constructor to write data into the file.

1. FileWriter fw=new FileWriter(String name);
2. FileWriter fw=new FileWriter(File f);

The above constructor will perform the overriding operation on the file I.e. the previous data will be deleted and new data will be carry by the file. However if we don’t want to delete the previous data then we can use the following operation to append the data with the existing data into the file. Following are the constructor to perform the append operation on the data.

1. FileWriter f=new FileWriter(String name, Boolean append)
2. FileWriter f=new FileWriter(file f, Boolean append)

Note: If the file mention in the constructor is not available in the directori then the constructor will create the file first and then perform the write operation on that file.

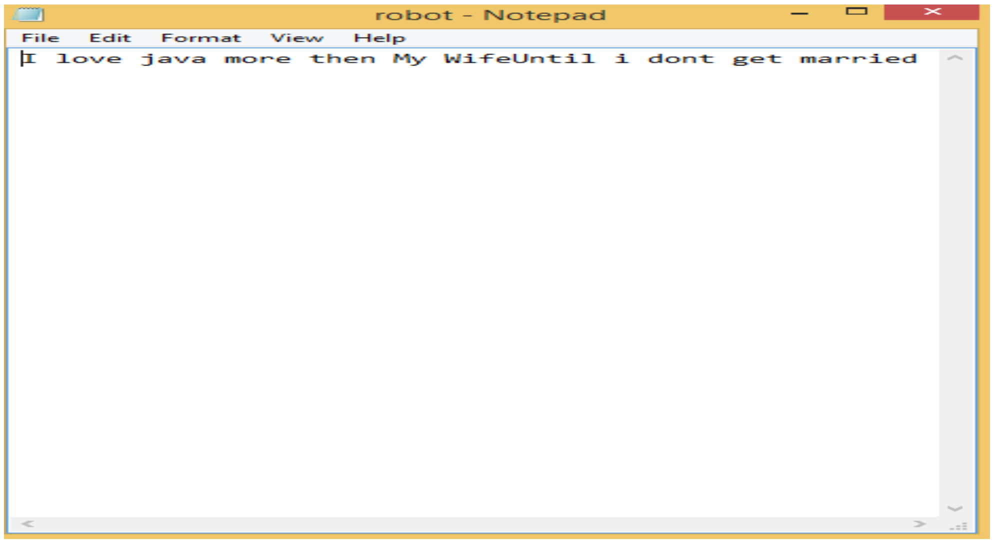
Following are the method in the File Writer to perform the operation.

1. Write(int ch): To write the single character into the file.
2. Write(char[] ch): To write the array of character into the file.
3. Write(String s): To write the string into the file.
4. Flush(): It will guarantee that all the character now write into the file.
5. Close(): To close the file.

Program to write the data into the file:

```
import java.io.*;

public class FileWrite1 {

    public static void main(String[] args) throws Exception
{
  FileWriter f=new FileWriter("C:\\Users\\saif\\Desktop\\robot.txt");
  f.write('I');
  f.write(" love java more then ");
  char ch[]={'M','y',' ','W','i','f','e'};
  f.write(ch);
  f.write("Until i dont get married");
  f.flush();
  f.close();
}
}
```

Output in file robot at position position C:\\Users\\saif\\Desktop\\robot.txt is



```
I love java more then My WifeUntil i dont get married
```

This program always perform append operation but if we want to perform append operation i.e. want to add some string in the existing string then we need to pass true argument in the program as FileWriter f=new FileWriter("C:\\Users\\saif\\Desktop\\robot.txt",true);

From this the new string will be upend to the existing string of the file.

File Reader:

File Reader is used to read the data from the file. Following are the constructor used in the file reader operation:

1. **FileReader f=new FileReader(File Name);**
2. **FileReader f=new FIleReader(file f);**

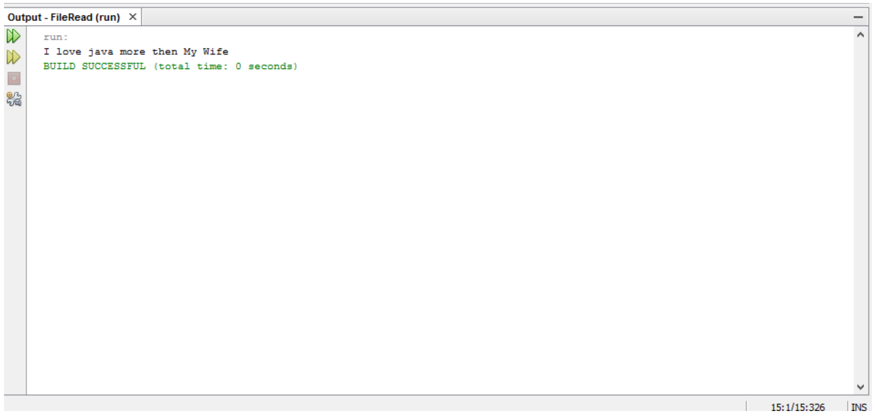
There are two method used in the file reader operation in the java.

1. **F.read();** It will read the Unicode value of the character in the file.bIf there is no next character them it will return -1.
2. **F.close();** It will close the file

Program to read the data from the file:

```
import java.io.*;

public class FileRead {

    public static void main(String[] args) throws Exception {

        FileReader f=new FileReader("C:\\Users\\saif\\Desktop\\robot.txt");

    int i=f.read();

    while(i!=-1)

    {

    System.out.print((char)i);

    i=f.read();

    }

    System.out.println();

    f.close();

    }

 }
```

Output:

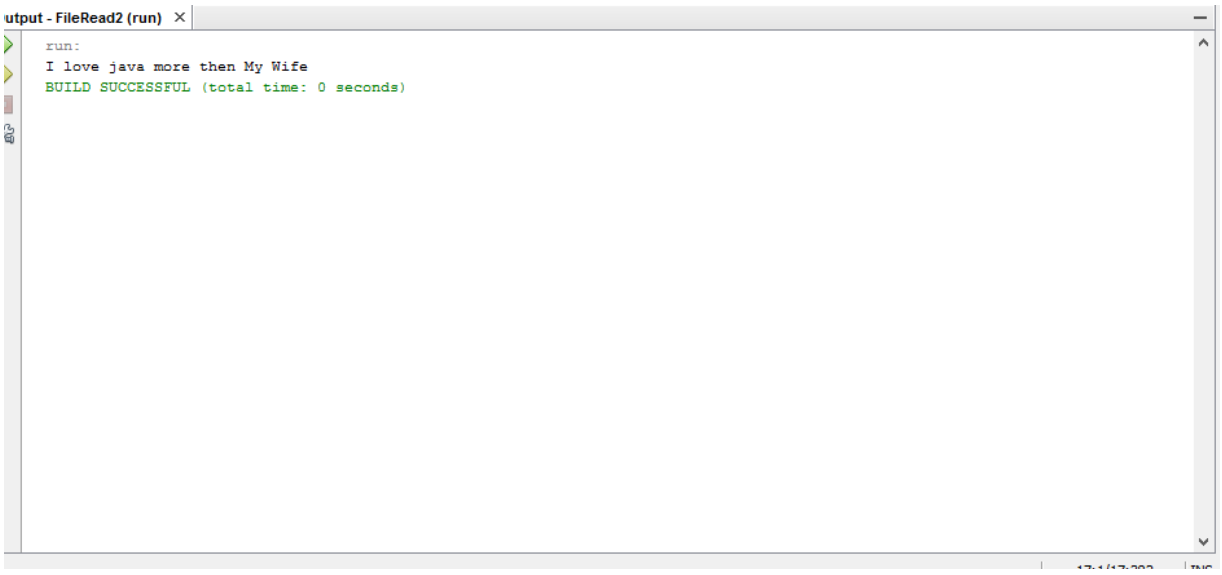


```
run:
I love java more then My Wife
BUILD SUCCESSFUL (total time: 0 seconds)
```

Program to read the data by the help of the array:

```
import java.io.*;
public class FileRead2
{
    public static void main(String[] args) throws Exception
{
        File f=new File("C:\\Users\\saif\\Desktop\\robot.txt");
        FileReader fr=new FileReader(f);
        char ch[]=new char[30];
         fr.read(ch);
         for(char ch1:ch)
         {
         System.out.print(ch1);
         }
         System.out.println();
}
}
```

Output:



```
run:
I love java more then My Wife
BUILD SUCCESSFUL (total time: 0 seconds)
```

File Reader and File Reader are nor recommended to use:

In File reader we have to insert \n operator for new line which is very from system to system.If we do not place the \n then the total data will be in the one line of the file.

And about the file reader the major problem us that we read the data character by the character not line by the line which is not convenient to the good program because it takes too long if we want to match the string in 100000 line of data file.

So they are the not recommended to use. So to overcome this we have buffered reader and the buffered writer instead of it.

Buffered Writer:

Buffered writer is used to read the data from the file.But rememeber in your mind that a buffer reader cn not communicate directly with the file as

BufferedWriter f=new BufferedWriter("robot.txt"); wrong

But   it can only communicate via some writer object only.

Constructor used in the buffered writer:

1. **Buffered Writer bw=new Buffered Writer(Writer w);**
2. **Buffered Writer bw=new Buffered Writer(Writer w.int buffersize);**

Method available in the buffered writer:

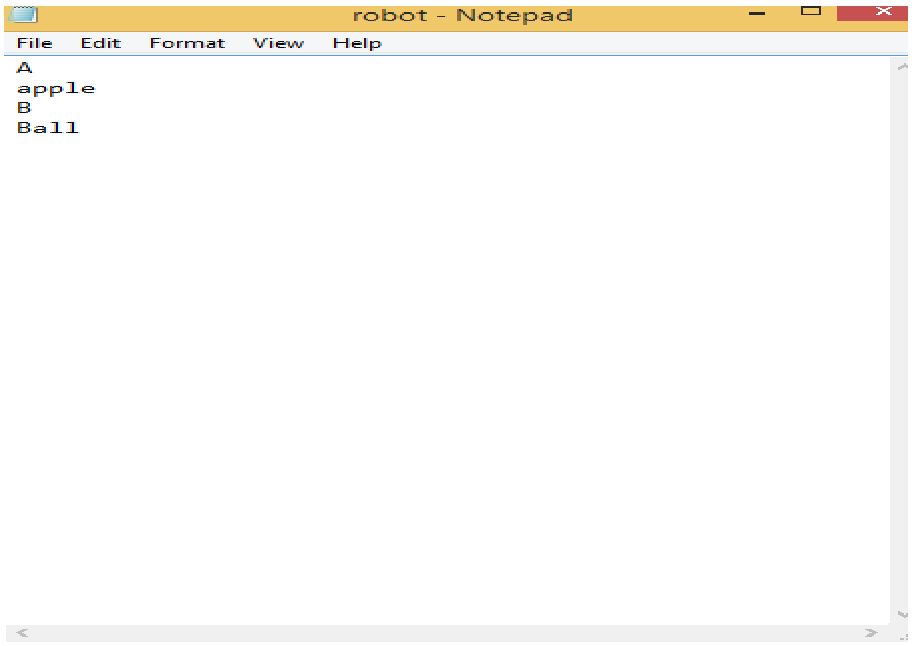
1. Write(int ch): To write the single character into the file.
2. Write(char[] ch): To write the array of character into the file.
3. Write(String s): To write the string into the file.
4. Flush(): It will guarantee that all the character now write into the file.
5. Close(): To close the file.
6. newline(): These is the new methid introduce as compare to the buffered writer to come the buffered reader problem.

Program to write a data into the File

```
import java.io.*;

public class FileBuffer1 {

   public static void main(String[] args) throws Exception {

      FileWriter f=new FileWriter("C:\\Users\\saif\\Desktop\\robot.txt");

      BufferedWriter bf=new BufferedWriter(f);

      bf.write(65);

      bf.newLine();
```

```
        char ch[]={'a','p','p','l','e'};
        bf.write(ch);
        bf.newLine();
        bf.write("B");
        bf.newLine();
        bf.write("Ball");
        bf.flush();
        bf.close();
    }
}
```

Output:



```
A
apple
B
Ball
```

When we perform the code bf.close() then the f.close() will be done automatically so we don’t need to write this code in our program.

Buffered Reader:

Buffered Reader is the best reader to read the data from the file and it read the data ine by line not character by character.It also can't communicate directly with the file but it can communicate via the some object.

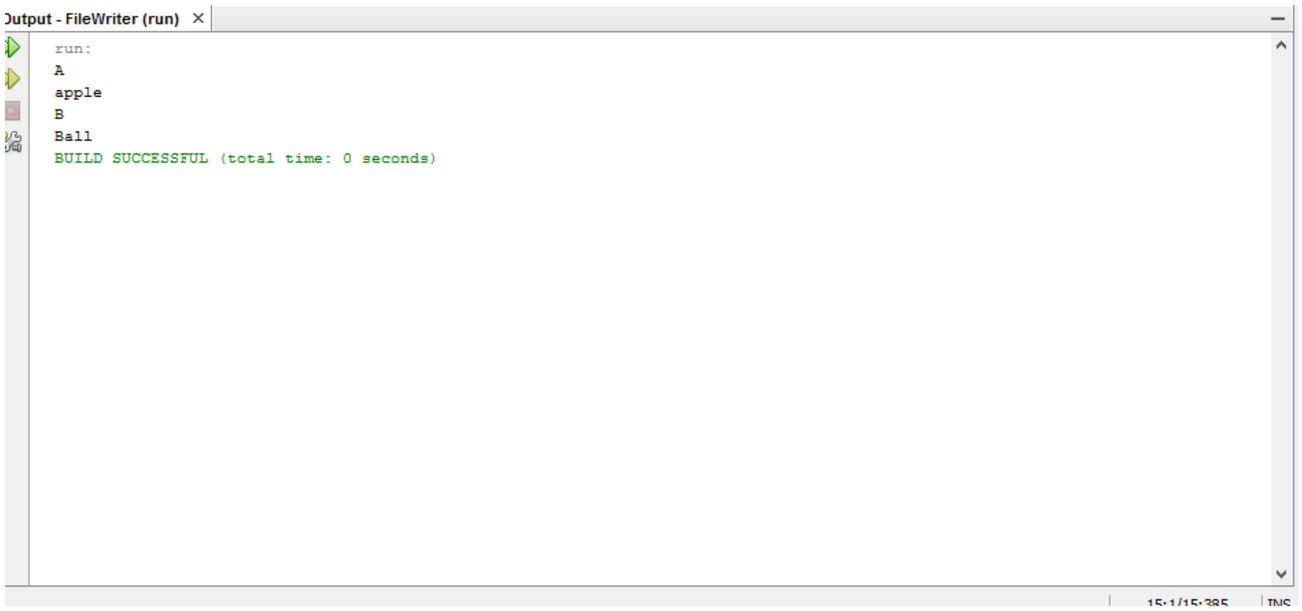
There are three method used in the Buffered reader operation in the java.

1. **Int read() or int read(ch[] ch);** It will read the Unicode value of the character in the file. If there is no next character them it will return -1.
2. **F.close();** It will close the file.
3. **String readline() It attempts to read next line from the file and returns it. If the next line is not available then it will return the null.**

Program to read the data from the file using buffered reader

```
import java.io.*;

public class FileWriter {

    public static void main(String[] args) throws Exception {

        FileReader f=new FileReader("C:\\Users\\saif\\Desktop\\robot.txt");

        BufferedReader bf=new BufferedReader(f);

        String s=bf.readLine();

       while(s!=null)

       {

       System.out.println(s);

       s=bf.readLine();

       }

       bf.close();

    }

 }
```

Output:



```
run:
A
apple
B
Ball
BUILD SUCCESSFUL (total time: 0 seconds)
```

There is no write method in java that can take the double and long argument.

Problem arises with the FileWriter and the BufferdWrite:

Apart from the new line operator \n or newline() the major problem with these two writer is they can print only character value but if you want to print the int , float , double even Boolean value then you have to pass it in the string which decrease the software performance. So to overcome this problem we have print define print writer which is the most powerful writer among these.

PrintWriter:

1. It reduce the problem of the new line operator like \n or newline() method now we can simply insert the new line along with the data to print in new line as println(data)
2. It also reduce the problem of writing character value by the buffered writer and file writer. Now we can write the any data with the help of the print such as int value, char value, double value and Boolean value so on.
3. Unlike buffered writer the print writer can also communicate directly with the file.

There is three constructor used in the print writer function:

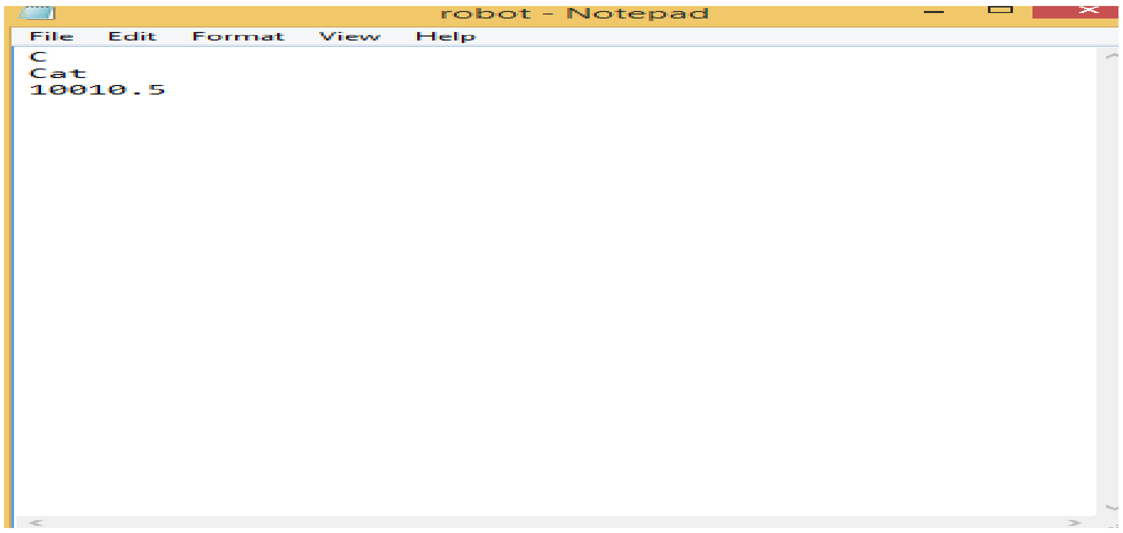
1. PrintWriter p=new PrintWriter(String s);
2. PrintWriter p=new PrintWriter(File f);
3. PrintWriter p=new PrintWriter(Writer w);

Program on print function in java:

```
import java.io.*;

public class Print1 {

   public static void main(String[] args) throws Exception
```

```
    {
     PrintWriter p=new PrintWriter("C:\\Users\\saif\\Desktop\\robot.txt");
     p.println('C');
     p.println("Cat");
     p.print(100);
     p.println(10.5);
     p.flush();
     p.close();
     }
}
```

OutPut:



```
C
Cat
10010.5
```

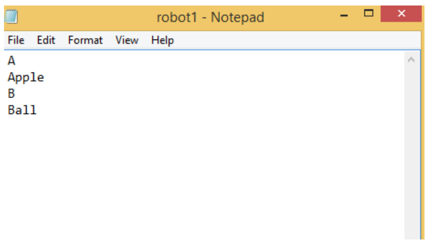
Remember first print action performed then new line will be added.

Program to merge two text file into a single text file:

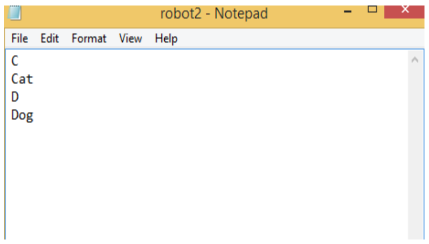
```
import java.io.*;
public class Ioprogram1
{
public static void main(String[] args) throws Exception
```

```
    {
        File f=new File("C:\\Users\\saif\\Desktop\\robot.txt");
        f.createNewFile();
        System.out.println(f.exists());
        PrintWriter p=new PrintWriter(f);
        FileReader f1=new FileReader("C:\\Users\\saif\\Desktop\\robot1.txt");
        BufferedReader bf1=new BufferedReader(f1);
        String s1=bf1.readLine();
        FileReader f2=new FileReader("C:\\Users\\saif\\Desktop\\robot2.txt");
        BufferedReader bf2=new BufferedReader(f2);
        String s2=bf2.readLine();
        while (s1!=null)
        {
        p.println(s1);
        s1=bf1.readLine();
        }
        while(s2!=null)
        {
        p.println(s2);
        s2=bf2.readLine();
        }
        p.flush();
        p.close();
    }
}
```

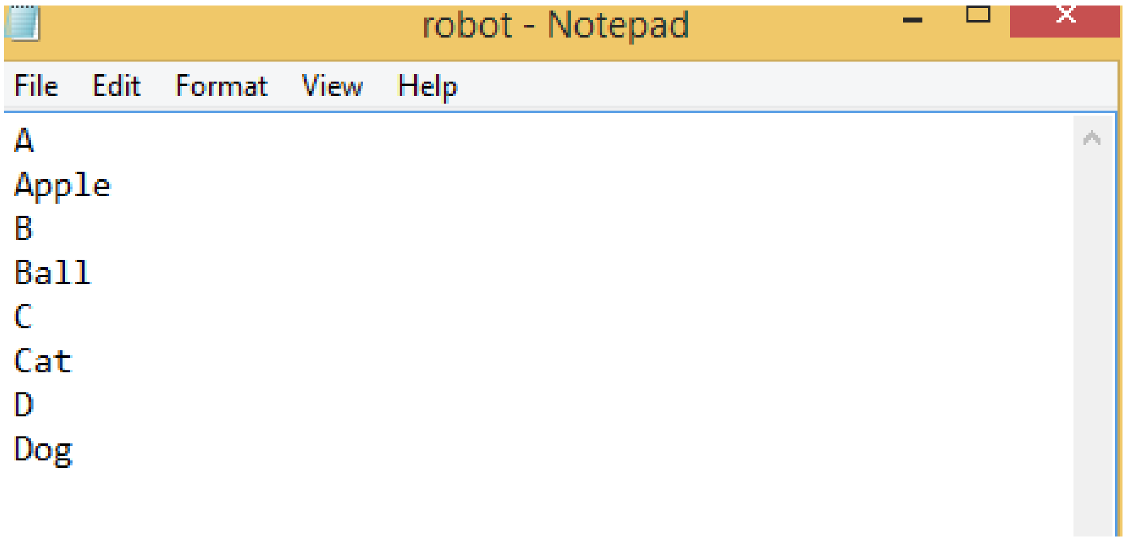
Output:



```
A
Apple
B
Ball
```



```
C
Cat
D
Dog
```



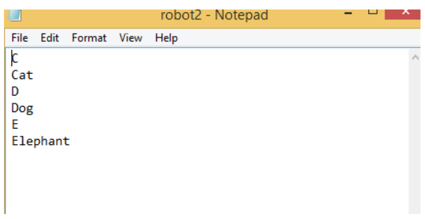
```
A
Apple
B
Ball
C
Cat
D
Dog
```

Program to merge two file into one file by alternative line:

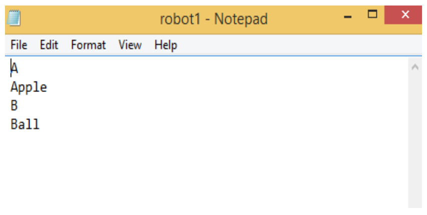
```
import java.io.*;
public class FileMerge {
   public static void main(String[] args) throws Exception {
      File f=new File("C:\\Users\\saif\\Desktop\\robotM.txt");
      f.createNewFile();
      System.out.println(f.exists());
      PrintWriter p=new PrintWriter(f);
      FileReader f1=new FileReader("C:\\Users\\saif\\Desktop\\robot1.txt");
      BufferedReader bf1=new BufferedReader(f1);
      FileReader f2=new FileReader("C:\\Users\\saif\\Desktop\\robot2.txt");
      BufferedReader bf2=new BufferedReader(f2);
```

```
        String s1=bf1.readLine();
        String s2=bf2.readLine();
        while(s1!=null || s2!=null)
        {
        if(s1!=null)
        {
        p.println(s1);
        s1=bf1.readLine();
        }
        if(s2!=null)
        {
        p.println(s2);
        s2=bf2.readLine();
        }
        }
        p.flush();
        p.close();
    }
}
```

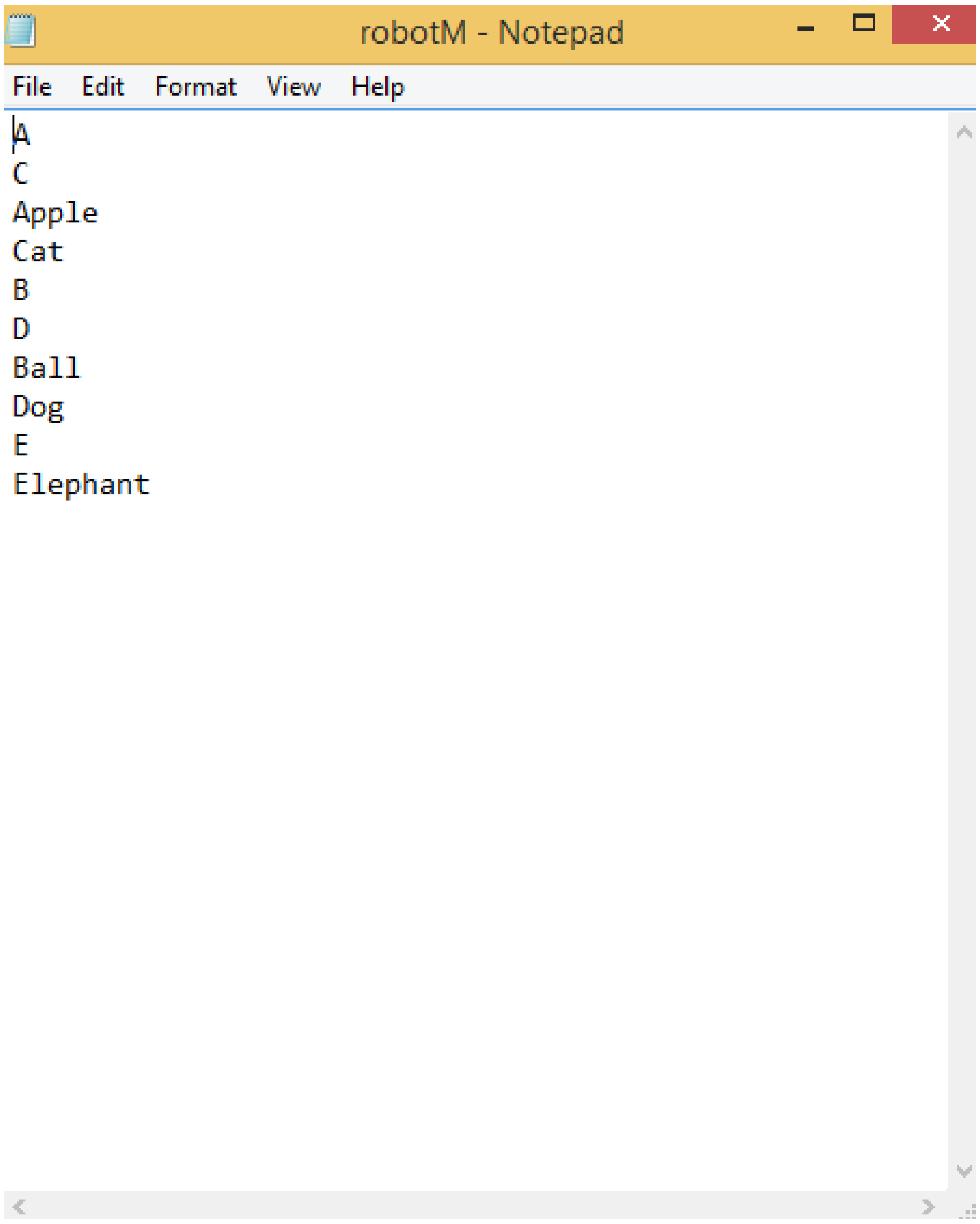
Output:



```
C
Cat
D
Dog
E
Elephant
```



```
A
Apple
B
Ball
```



```
A
C
Apple
Cat
B
D
Ball
Dog
E
Elephant
```

Multithreading in Java:

In java thread means smallest independent unit consisting of the code to perform different operation at the same time with the help of switching the task between them by the processor at the high speed.

In java we use the multithreading to achieve the multitasking. At a time only one task is executed inside the process.

Advantage of the thread in the java:

4. Smallest independent unit so not disturb or block other task.
5. It saves times because we can perform different operation at the same time.
6. It saves the memory because it shares the same memory area.
7. There can be multiple processes inside a thread but one process can have the multiple threads.

A thread is executed inside the process in any of the following five cycle in order to complete the task.These four states are

1. New State
2. Runnable state
3. Running state
4. Waiting state
5. Dead State

New State:

A process is in the ready state when it is created. It is invoked by the method start ().

Runnable state:

A thread is in the runnable state after invoked the method start but it is not assigning to the thread scheduler to running.

Running state

When the thread scheduler selected it the thread is in the running state.

Wait State

Wait

When a thread is not eligible to run but it is wait for run then it is said to be in the wait state.

Dead state:

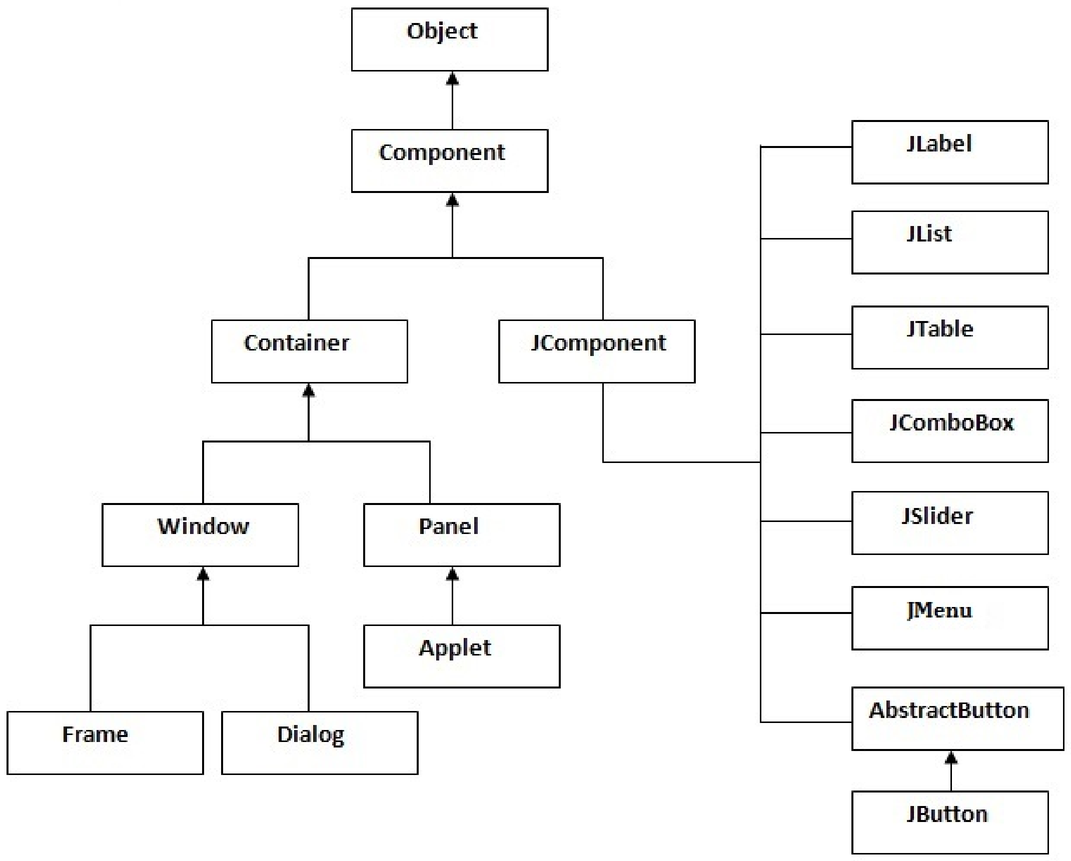
When the thread run method exists it is said to be in the dead state.

How thread is created in the java:

Swing :

Java swing is usd to develop the window based application using java.Java swing is platfoerm dependent and the light weight tool for developing desktop application such as NetBeans etc. The class of swing is provided in javax.swing package in order to provide the java swing API such as JFrame, JButton , JTextfield , JRadioButton etc. It also provide the more powerful contain such as table , list , colorchooser etc.

Hierachy of java swing class is given below



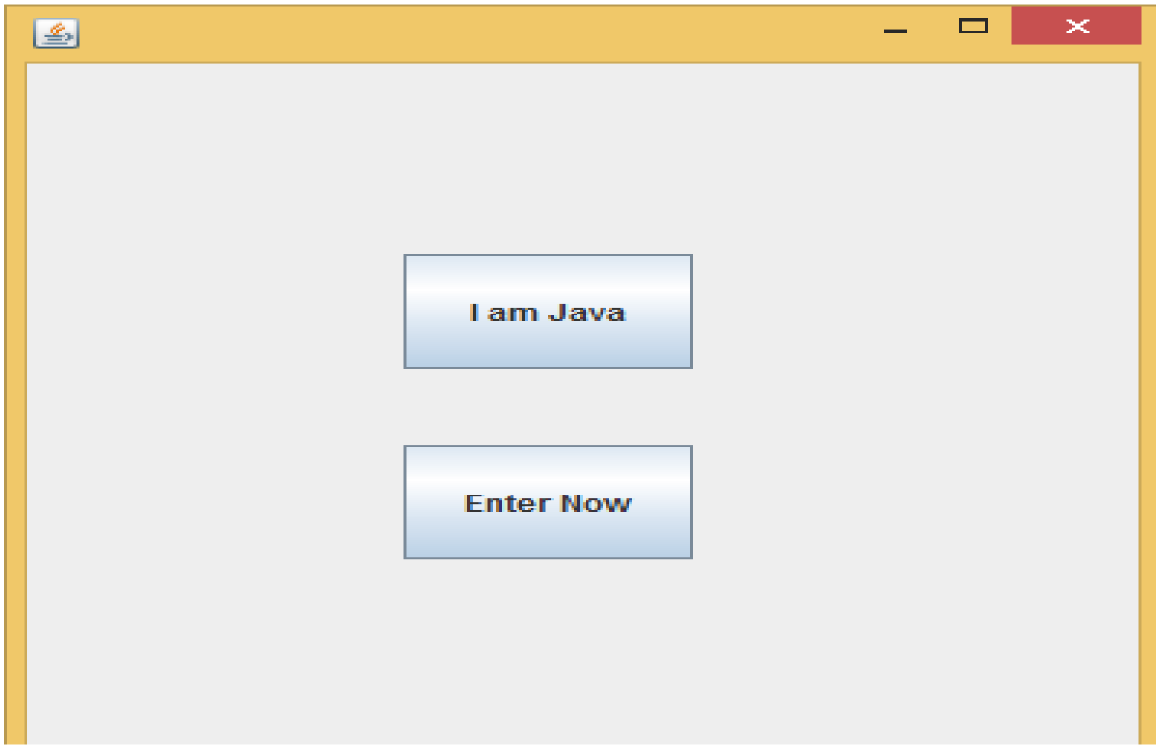
Implementation of swing program:

Java Swing application can be implanted in two ways:

3. By creating the object of the frame class
4. By extending the frame class.

Program of By Creating the object of the frame class:

```
import javax.swing.*;

public class SwingMe

{
  public static void main(String[] args)
  {
    JFrame j=new JFrame();
    j.setVisible(true);
    j.setSize(400,400);
    j.setLayout(null);
    JButton b=new JButton("I am Java");
    j.add(b);
    b.setBounds(130,100, 100, 60);
    JButton c=new JButton("Enter Now");
    j.add(c);
    c.setBounds(130,200, 100, 60);
  }
}
```



Important lines of the program:

4. Import javax.swing.*; is the header file which contain the classes of button , textfields , list etc of swing application.
5. Create the bobject of the frame by JFrame. JFrame is the new window of the application.
6. The Frame will not visible on the screen until you define the frame as j.setVisible(true);
7. Set the frame size from j.setSize(400,400); which is for width and height of the frame.
8. Create the object of the button and set the text field on it by JButton b=new JButton("I am Java");
9. Add the button on the frame with location parameter as j.add(b);
10. Set the size and boundry of the button by b.setBounds(130,100, 100, 60);

Program for the swig using the constructor:

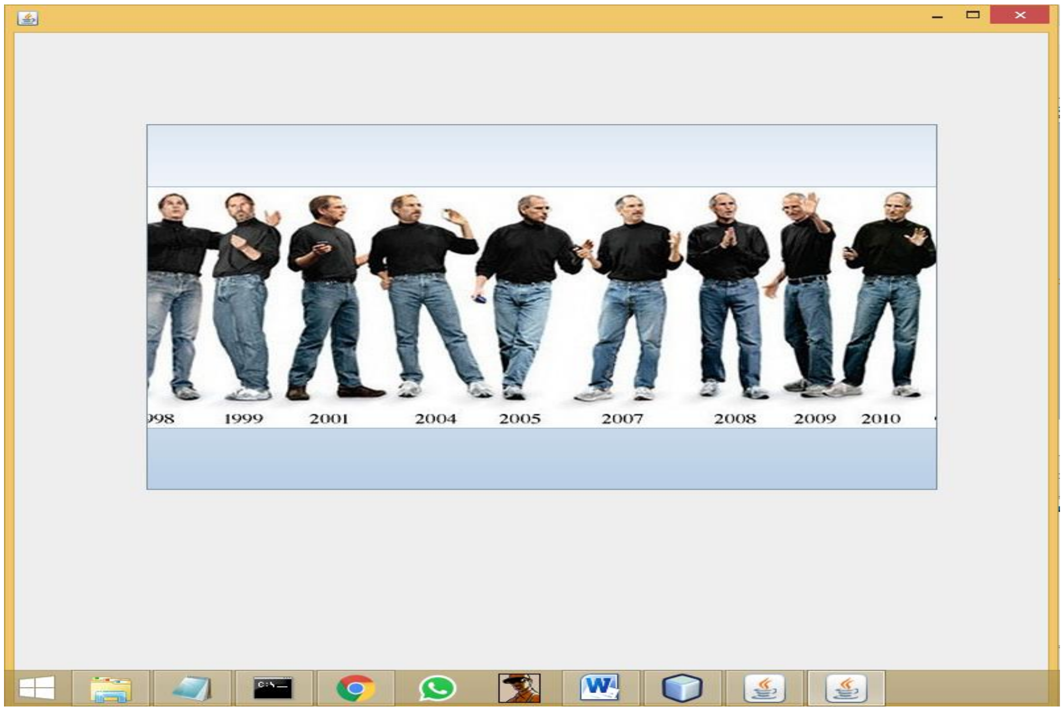
JButton:

JButton is used to create the button that has platform independent implementation .We can create the button with no text and icon or with the some text or with the some icon.

Program to demonstrate JButton with the icon.

```
import javax.swing.*;

public class ButtonImage

{

ButtonImage()

  {

 JFrame j=new JFrame();

 j.setVisible(true);

 j.setSize(800,800);

 j.setLayout(null);

 JButton b=new JButton(new ImageIcon("C:\\Users\\saif\\Desktop\\sj.jpg"));

 j.add(b);

 b.setBounds(100, 100, 600,400);

  }

public static void main(String[] args)
{
new ButtonImage();
}
}
```

OutPut:



JRadioButton:

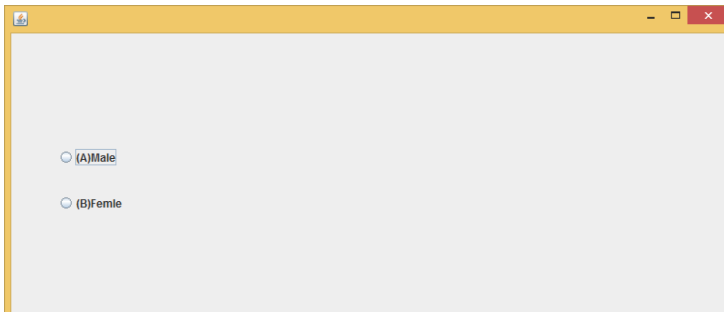
Java Radio Button is used to select one button among two or more button>It is widely used in MCQ question.

```
import javax.swing.*;
public class RadioButton {
public RadioButton()
{
JFrame j=new JFrame();
j.setSize(800,800);
j.setVisible(true);
j.setLayout(null);
JRadioButton r1=new JRadioButton("(A)Male");
JRadioButton r2=new JRadioButton("(B)Femle");
r1.setBounds(50,100,120,70);
r2.setBounds(50,150,120,70);
ButtonGroup bg=new ButtonGroup();
bg.add(r1);
bg.add(r2);
j.add(r1);
j.add(r2);
}
   public static void main(String[] args) {
       new RadioButton();
   }
}
```

Program Headlines:

ButtonGroup class can be used to group multiple buttons so that at a time only one group can e selected.

OutPut:



JTextArea:

JTextArea is used to create the text box with specified number of rows and column. It is used to display the plain text only.

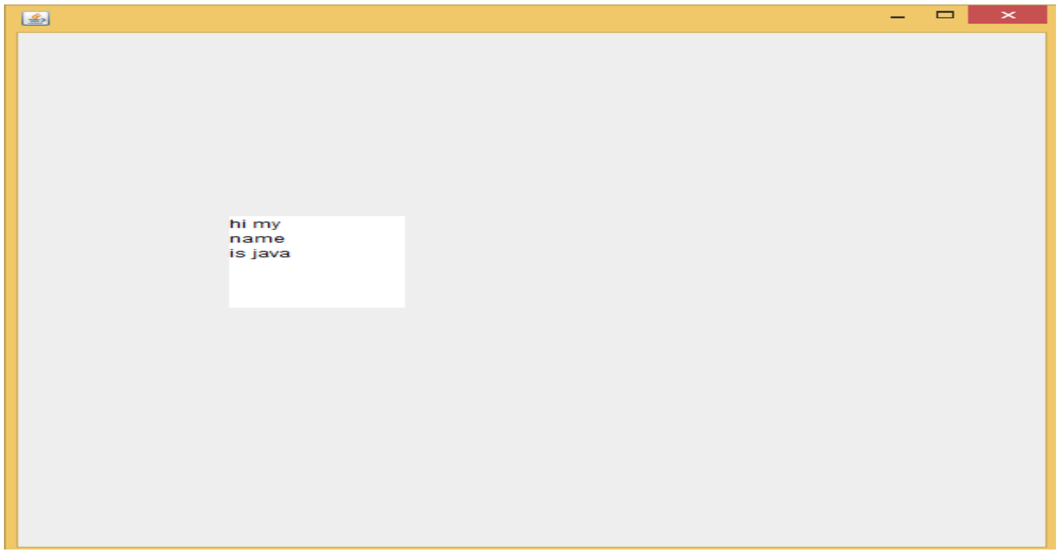
A JTextArea() can be used in the following way:

5. **JTextArea():** It will create the text area without any text.
6. **JTextArea("String S"):** It will create the text with the string word.
7. **JTextArea(int row, int col):**Create the TextArea qith given number of rows and columns.
8. **JTextArea(String s,int row, int col):**Create the text area with string on specified rows and columns.

Program to demonstrate the JTextArea example

```
import java.awt.Color;

import javax.swing.*;

public class Textme {

   JFrame f;

   JTextArea jt;

   public Textme()

   {

   f=new JFrame();

   f.setSize(600,600);

   f.setVisible(true);

   f.setLayout(null);

   jt=new JTextArea(100,100);

   f.add(jt);

   jt.setBounds(120,200,100,100);

   jt.setBackground(Color.white);

   jt.setForeground(Color.black);

   }

   public static void main(String[] args) {

   new Textme();

   }

 }
```

OutPut:


hi my
name
is java

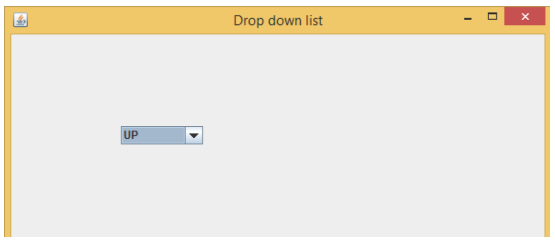
JComboBox():

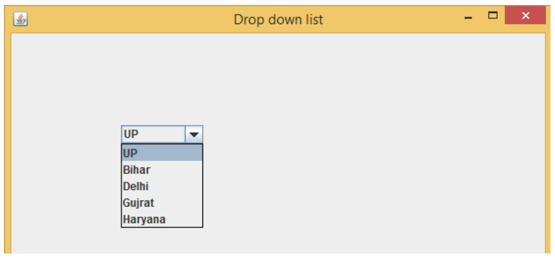
Java combo box is used to select the item form one or more item.like to choose a country among many or choose a state , religion from the give list we used combo box.

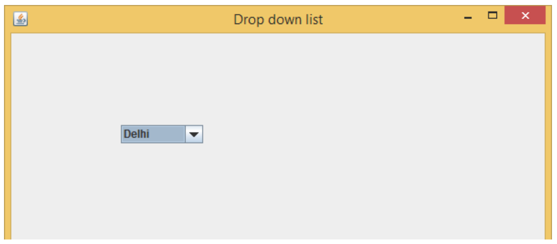
Program to demonstrate JComboBox();

```
import javax.swing.*;

public class Dropdown {

   JFrame f;

   JComboBox jc;

   public Dropdown()

   {

       f=new JFrame("Drop down list");

      f.setSize(600,600);

      f.setVisible(true);

      f.setLayout(null);

      String state[]={"UP","Bihar","Delhi","Gujrat","Haryana"};

      jc=new JComboBox(state);

      jc.setBounds(120,100,90,20);

      f.add(jc);
```

```
    }
  public static void main(String[] args) {
  new Dropdown();
  }
}
```







JColorChooser class:

JColorChooser class is used to create the pop up of the JColorChooser dialog box so that the user can select at any color and color the desired area.

Progran to demonstrate the JColorChooser class:

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class Colorchoose extends JFrame implements ActionListener
{
 JFrame f;
 Container c;
 JButton b;
 public Colorchoose()
   {
 setSize(600,600);
 //setLayout(null);
 setVisible(true);
 c=getContentPane();
 c.setLayout(new FlowLayout());
 b=new JButton("color");
 b.addActionListener(this);
 c.add(b);
        }
 public void actionPerformed(ActionEvent e)
 {
 Color Initialcolor=Color.RED;
 Color color=JColorChooser.showDialog(this,"Select a coor",Initialcolor);
 c.setBackground(color);
 }

    public static void main(String[] args)
    {
        new Colorchoose();
    }
```

}

Hint Tag:

3. c=getContentPane(); **we know that the content pane is the layer of the container , so in the container with the help of the** c=getContentPane(); we makes the content pane available so that you can apply a method to it.For example to change the color of the content pane we must first use the getContentPane(); method.
4. In simple words a container is compose of the several transparent layer and a layer whch is used to hold the object such as button, box, list etc are called the content pane. With the help of getContentPane we retrieve the layer to add the object on it.

JSlider Class:

JSlider class is used to create the slider which is used to choose a value between the specific ranges.

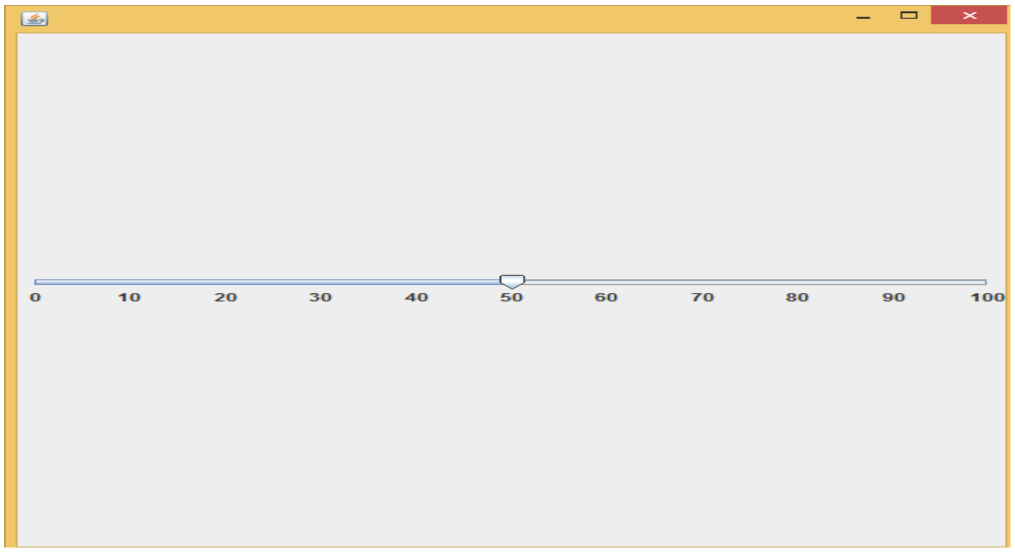
Java Program to demonstrate the JSliderClass

```
import javax.swing.*;

public class Jsavalider {

JFrame f;

JSlider js;

public  Jsavalider()

{

f=new JFrame();

f.setVisible(true);

f.setSize(600,600);

js=new JSlider(JSlider.HORIZONTAL,0,100,50);

f.add(js);

js.setMajorTickSpacing(10);

js.setPaintLabels(true);


}
public static void main(String[] args) {
        new  Jsavalider();
    }


}
```

Hints Tags:

2. JSlider(JSlider.HORIZONTAL,0,100,50); creates the slider on horizontal axis with the minimum value 0 and maximum value 100, the initial value of the cursor is set on the 50 by this function.
3. setMajorTickSpacing(10); is used to tick a mark after every 10 number between the space 0-100.
4. setPaintLabels(true); is used to paint he value.

Output:



Difference between the JFrame and the JPanel in the Java:

JFrame is used to create the window which have a minimize , maximize and a close button while JPanel not include these button but used as same for holding the JButton, JSlider etc.

Canvas in Java:

With the help of the canvas we create the blank rectangular area on which we can draw the image or do soe input event from the user. In order to fulfill the functionality the application call must be the sub class of the canvas.

Canvas is the part of the abstract windowing tool.

Data structure

Stack:

A stack is the linear data structure in which the element add or remove only at one end which is called the top of the stack. When the item is added into the stack it is called the PUSH operation and when the item is removed from the stack it is called the pop operation.

The stack operation is based on the LIFO principal which means Last in First Out.

For Example: Let us consider the stack of the Pennies or stack of the books in all these you know you can add or remove the item only from the top not from the middle nor from the end only from the top.

Application:

1. The Undo operation in any of the software or ctrl+z operation is always done by the help of the stack.
2. All the Polish notation problem are like postfix problem and prfiz problem of mathematics is computer are solved by the help if the stack.

Program to implement stack operation Push , Pop , Size and Peek Value:

```
import java.util.*;

class StackP

{

 int a[]=new int[10];

 int top, len, size;

 public StackP(int n)

    {

     size=n;

     top=-1;

     len=0;

    }

public int size()

{
```

```
if(empty())
throw new NoSuchElementException("Underflow condition");
return len;
}
public boolean empty()
{
return top==-1;
}
public int peek()
{
 if(empty())
  throw new NoSuchElementException("Underflow Exception");
        return a[top];
}
public void push(int i)
    {
     if(top+1>=size)
        {
         System.out.println("Overflow condition");
        }
else
       {
        a[++top]=i;
        len++;
       }
    }
public int pop()
{
```

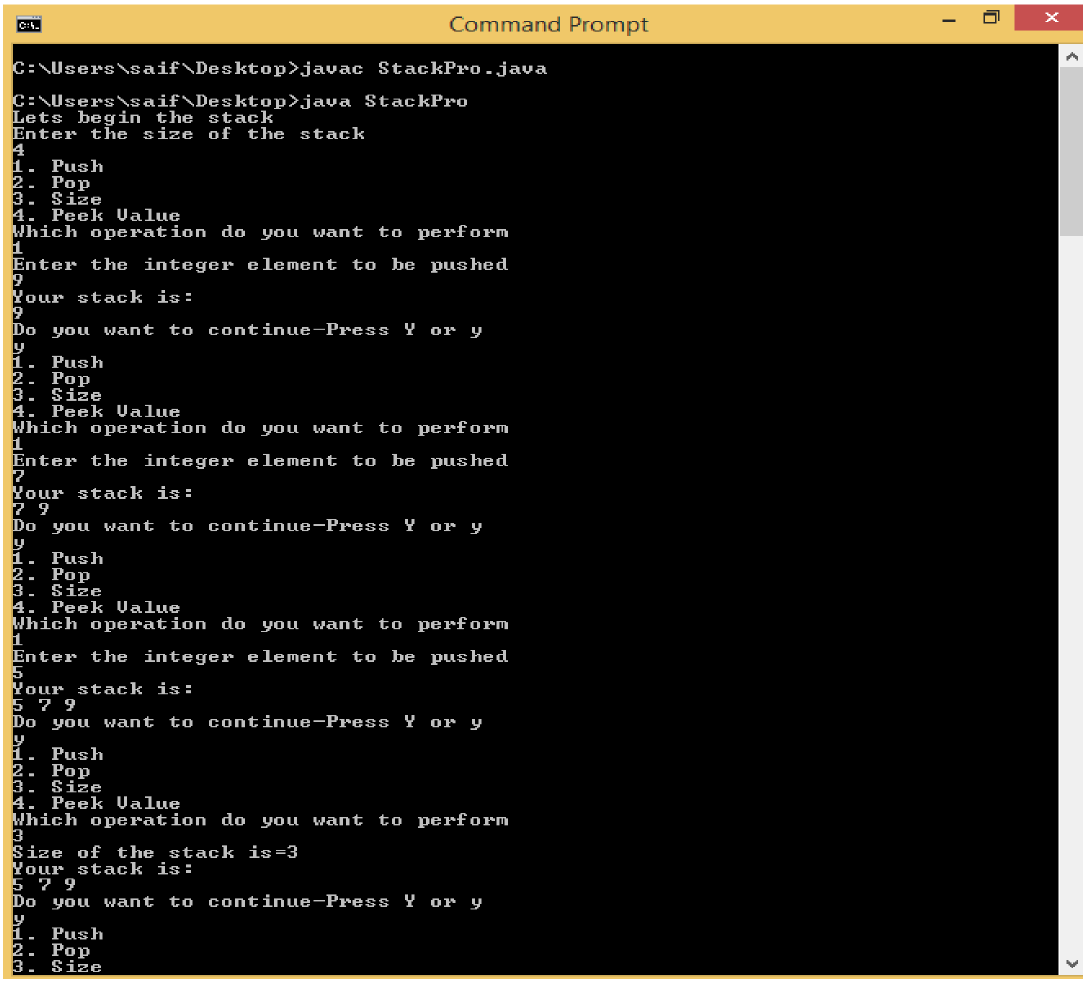
```
if(empty())
throw new NoSuchElementException("Underflow condition");
len--;
return a[top--];
}

public void display()
   {
   if(len==0)
     {
      System.out.print("Stack is empty");
     }
   else
     {
      System.out.println("Your stack is:");
      for(int i=top;i>=0;i--)
       {
        System.out.print(a[i]+" ");

       }
     }
   }

}



class StackPro
   {
```

```
   public static void main(String[] args)
    {
     Scanner in=new Scanner(System.in);
     System.out.println("Lets begin the stack");

     System.out.println("Enter the size of the stack");
     int n=in.nextInt();
     StackP sp=new StackP(n);
     char ch;
     do
       {
        System.out.println("1. Push");
        System.out.println("2. Pop");
        System.out.println("3. Size");
        System.out.println("4. Peek Value");
        int choose;
        System.out.println("Which operation do you want to perform");
        choose=in.nextInt();
        switch(choose)
         {
            case 1:
            System.out.println("Enter the integer element to be pushed");
            try
            {
             sp.push(in.nextInt());
            }
            catch(Exception e)
            {
```

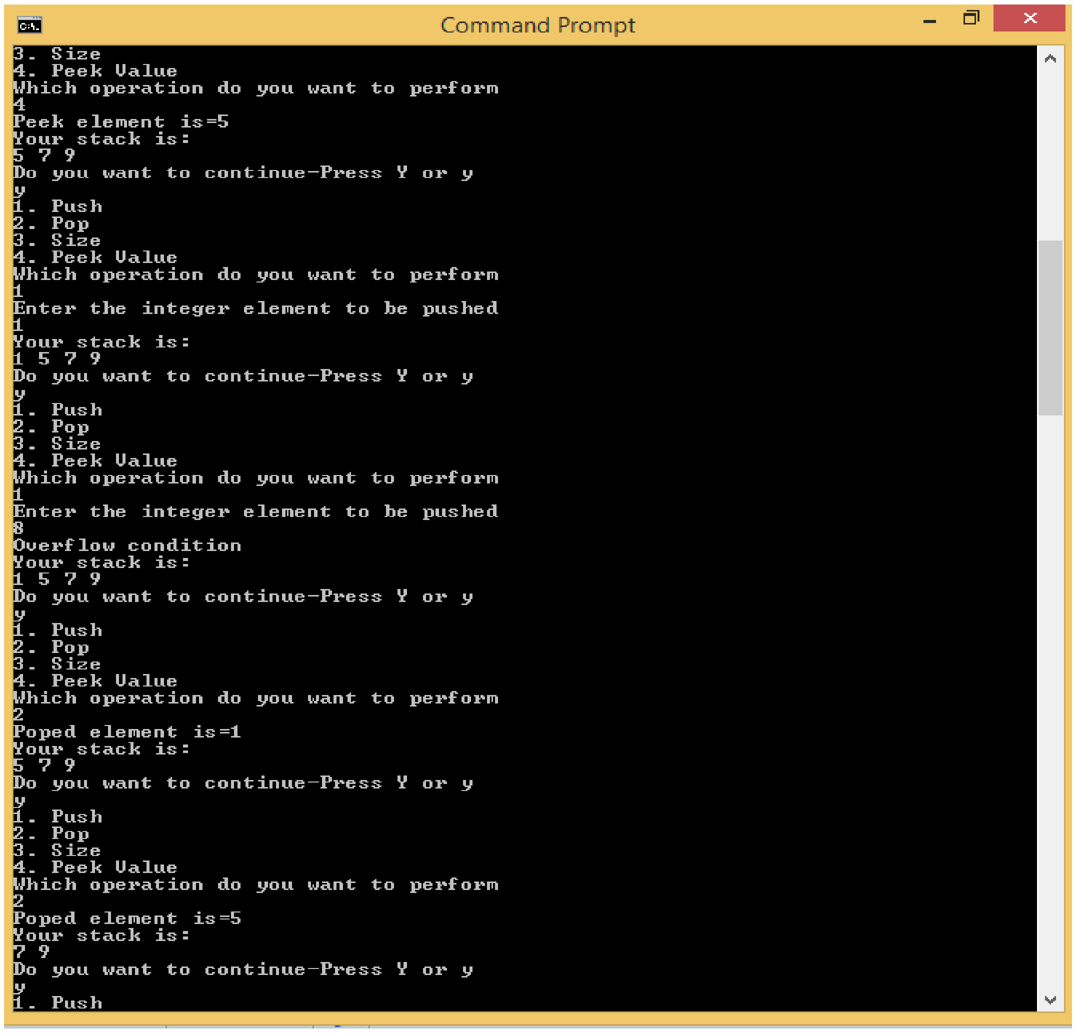
```
  System.out.println("Error:"+e.getMessage());
}
break;
case 2:
try
{
 System.out.println("Poped element is="+sp.pop());
}
catch(Exception e)
{
System.out.println("Error:"+e.getMessage());
}
break;
case 3:
try
{
 System.out.println("Size of the stack is="+sp.size());
}
catch(Exception e)
{
System.out.println("Error:"+e.getMessage());
}
break;
case 4:
try
{
 System.out.println("Peek element is="+sp.peek());
}
```

```
            catch(Exception e)
            {
            System.out.println("Error:"+e.getMessage());
            }
            break;

         }
    sp.display();
System.out.println("");
System.out.println("Do you want to continue");
ch=in.next().charAt(0);
}
while(ch=='y' || ch=='Y');
}
}
```

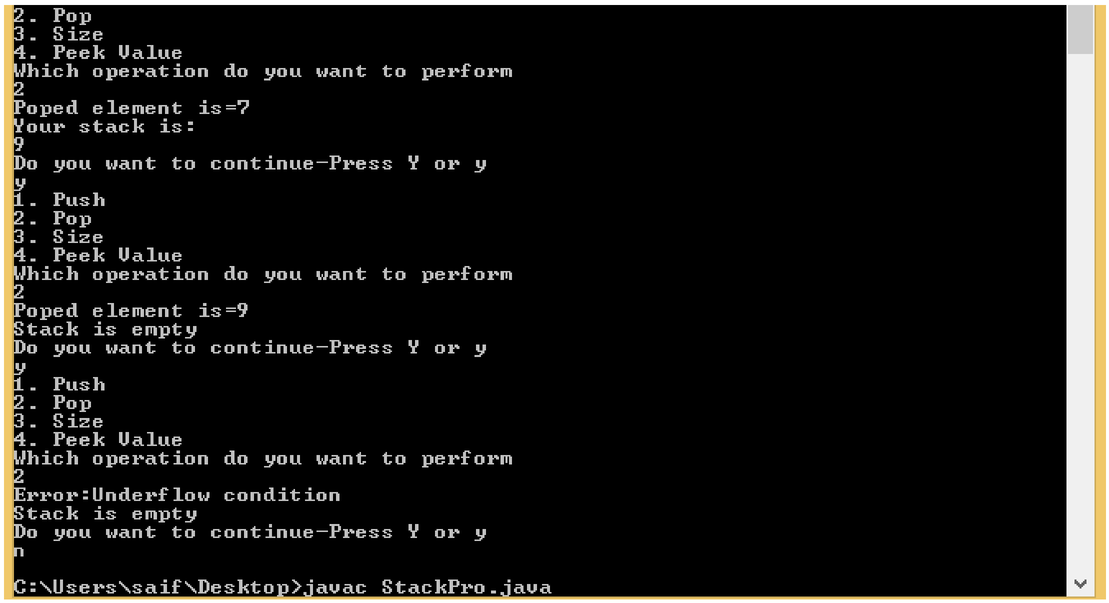
Output:



```
C:\Users\saif\Desktop>javac StackPro.java

C:\Users\saif\Desktop>java StackPro
Lets begin the stack
Enter the size of the stack
4
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
1
Enter the integer element to be pushed
9
Your stack is:
9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
1
Enter the integer element to be pushed
7
Your stack is:
7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
1
Enter the integer element to be pushed
5
Your stack is:
5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
3
Size of the stack is=3
Your stack is:
5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
```



```
3. Size
4. Peek Value
Which operation do you want to perform
4
Peek element is=5
Your stack is:
5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
1
Enter the integer element to be pushed
1
Your stack is:
1 5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
1
Enter the integer element to be pushed
8
Overflow condition
Your stack is:
1 5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
2
Poped element is=1
Your stack is:
5 7 9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
2
Poped element is=5
Your stack is:
7 9
Do you want to continue-Press Y or y
y
1. Push
```

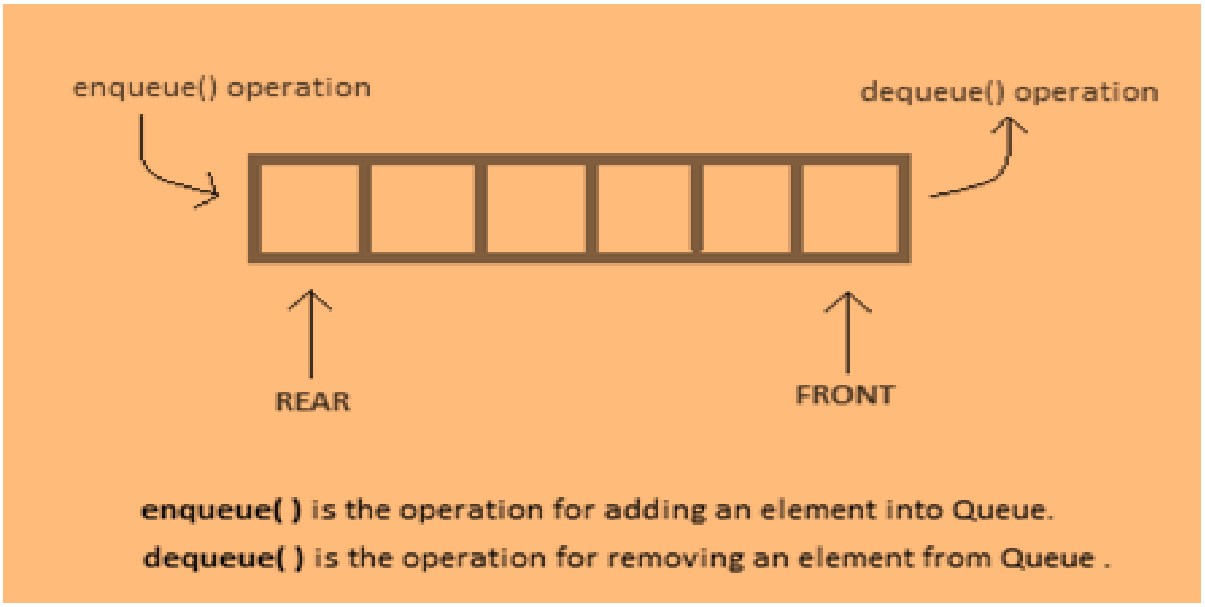


```
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
2
Poped element is=7
Your stack is:
9
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
2
Poped element is=9
Stack is empty
Do you want to continue-Press Y or y
y
1. Push
2. Pop
3. Size
4. Peek Value
Which operation do you want to perform
2
Error:Underflow condition
Stack is empty
Do you want to continue-Press Y or y
n

C:\Users\saif\Desktop>javac StackPro.java
```

Queue:

A queue is abstract data structure in which has two ends. The head of the queue is called front end and the tail of the queue is called the rear end. The data in the queue is inserted at its rear end and removed or delete from the front end. The queue work on the principal of the fifo (First in & First out). The data which is inserted will be removed first on the removal operation.

To add data into queue is called the enqueue operation and the removal of the data from queue is called the dequeue operation.


enqueue() operation
dequeue() operation
REAR
FRONT
enqueue( ) is the operation for adding an element into Queue.
dequeue( ) is the operation for removing an element from Queue .

Front indicate the top most position of the element in the queue.

Rear indicates the last inserted element position of the queue.

In queue element is always inserted at rear position.

Application of the queue:

9. **Serving the resource to the available entity into the queue.**
10. **Attend the waiting call in the call center or customer care**

Java program to demonstrate the queue operation:

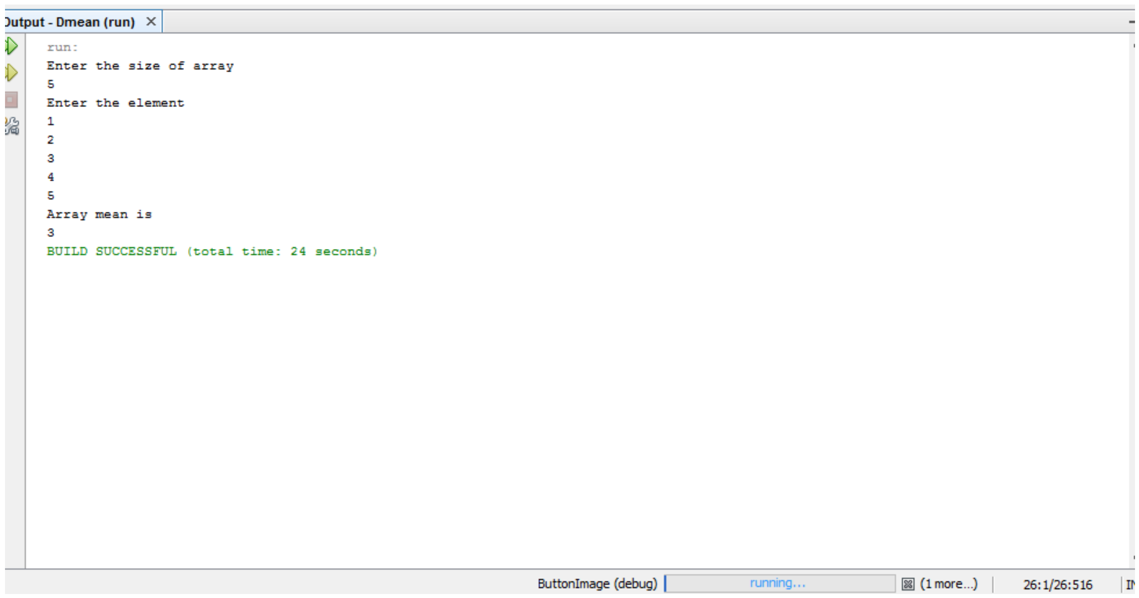
Data structure Program:

1. **WAP to find mean of n numbers using array.**

```
import java.util.*;
```

```
public class Dmean
{
    public static void main(String[] args)
   {
       int a[]=new int[10];
       int n,s=0,dm;

    Scanner in=new Scanner(System.in);
    System.out.println("Enter the size of array");
    n=in.nextInt();
    System.out.println("Enter the element");
    for(int i=0;i<n;i++)
    {
    a[i]=in.nextInt();
    }
    System.out.println("Array mean is");
    for(int i=0;i<n;i++)
    {
    s=s+a[i];
    }
    System.out.println(s/n);
   }

}
```

Output:



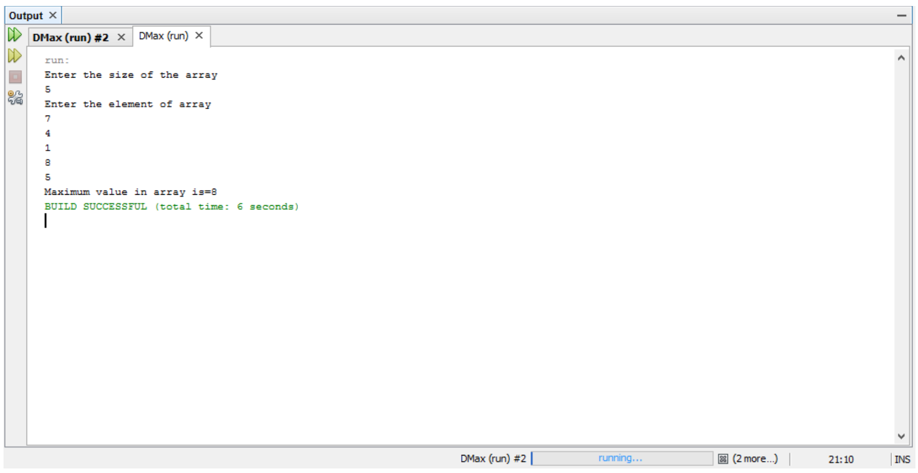
```
run:
Enter the size of array
5
Enter the element
1
2
3
4
5
Array mean is
3
BUILD SUCCESSFUL (total time: 24 seconds)
```

2. WAP to find largest of n number using array.

```
import java.util.*;
public class DMax {

    public static void main(String[] args) {
        int a[]=new int[10];
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the size of the array");
        int n=in.nextInt();
        System.out.println("Enter the element of array");
        for(int i=0;i<n;i++)
        {
        a[i]=in.nextInt();
        }
        System.out.print("Maximum value in array is=");
        int max=a[0];
        for(int i=1;i<n;i++)
        {
```

```
        if(a[0]<a[i])
        {
        max=a[i];
        }
        }
         System.out.print(max);
         System.out.println("");
    }
}
```

Output:

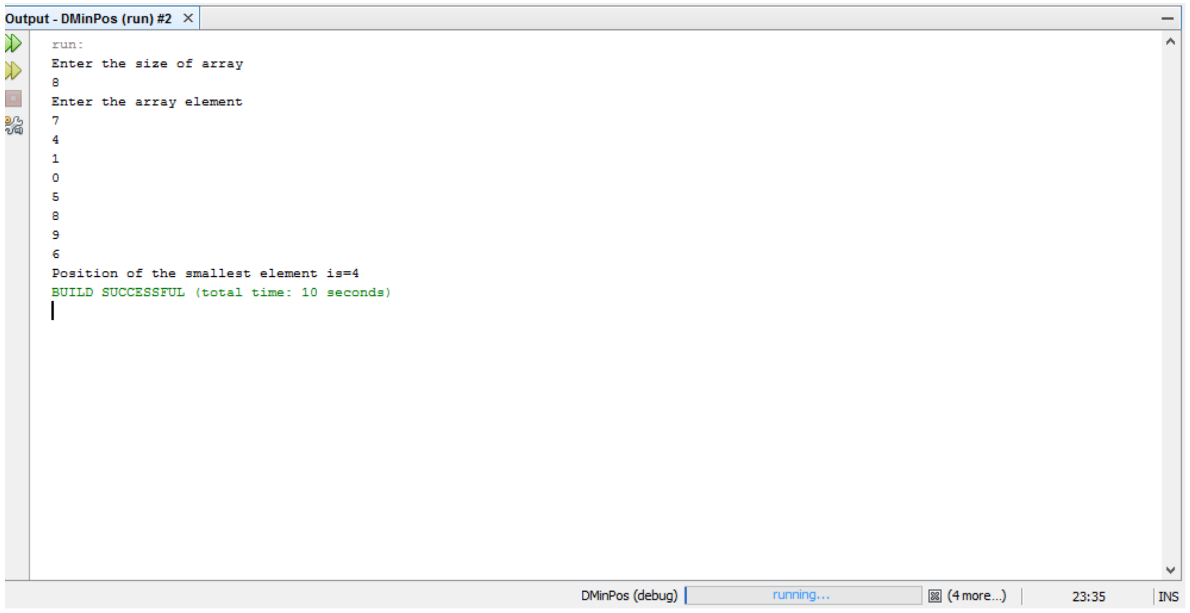


```
run:
Enter the size of the array
5
Enter the element of array
7
4
1
8
5
Maximum value in array is=8
BUILD SUCCESSFUL (total time: 6 seconds)
```

3. WAP in to find position of smallest element in array.

```
import java.util.*;
public class DMinPos {
    public static void main(String[] args) {
        int n;
        int a[]=new int[10];
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the size of array");
        n=in.nextInt();
        System.out.println("Enter the array element");
        for(int i=0;i<n;i++)
           {
        a[i]=in.nextInt();
           }
        int min=0;
        System.out.println("Position of the smallest element is=");
        for(int i=1;i<n;i++)
           {
        if(a[min]>a[i])
            {
        min=i;
            }
          }
        System.out.print(min+1);
        System.out.println();
         }
}
```

Output:


Output - DMinPos (run) #2

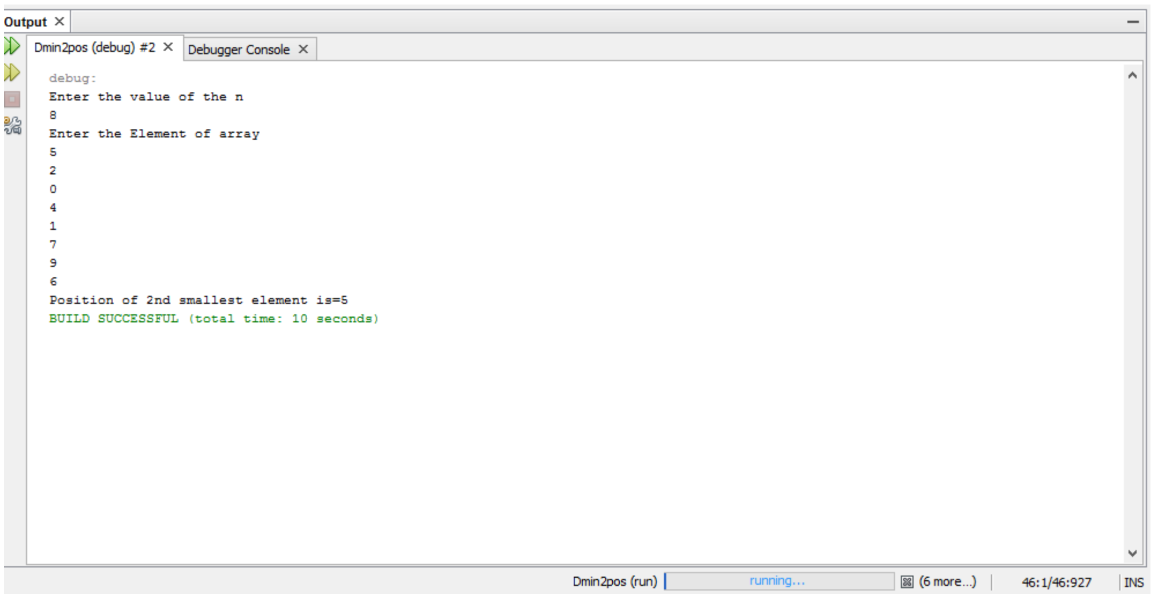
```
run:
Enter the size of array
8
Enter the array element
7
4
1
0
5
8
9
6
Position of the smallest element is=4
BUILD SUCCESSFUL (total time: 10 seconds)
```

DMinPos (debug) running... (4 more...) 23:35 INS

4. WAP in c to find position of second smallest element in array.

```
import java.util.*;
public class Dmin2pos {

    public static void main(String[] args) {
        int a[]=new int[10];
        int n;
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the value of the n");
        n=in.nextInt();
        System.out.println("Enter the Element of array");
        for(int i=0;i<n;i++)
        {
        a[i]=in.nextInt();
        }
        System.out.print("Position of 2nd smallest element is=");
        int min1=0;
        int min2=1;
        for(int i=2;i<n;i++)
        {
        if(a[min1]>a[i])
        {
        min1=i;
        }
        }
        for(int i=0;i<n;i++)
        {
        if(i!=min1)
        {
        if(a[min2]>a[i])
        {
        min2=i;
        }
        }
        }
        if(a[min1]<a[min2])
        {
        System.out.println(min2+1);
        }
        else
        {
        System.out.println(min1+1);
        }
    }

}
```

Output:



Output ×
Dmin2pos (debug) #2 × Debugger Console ×

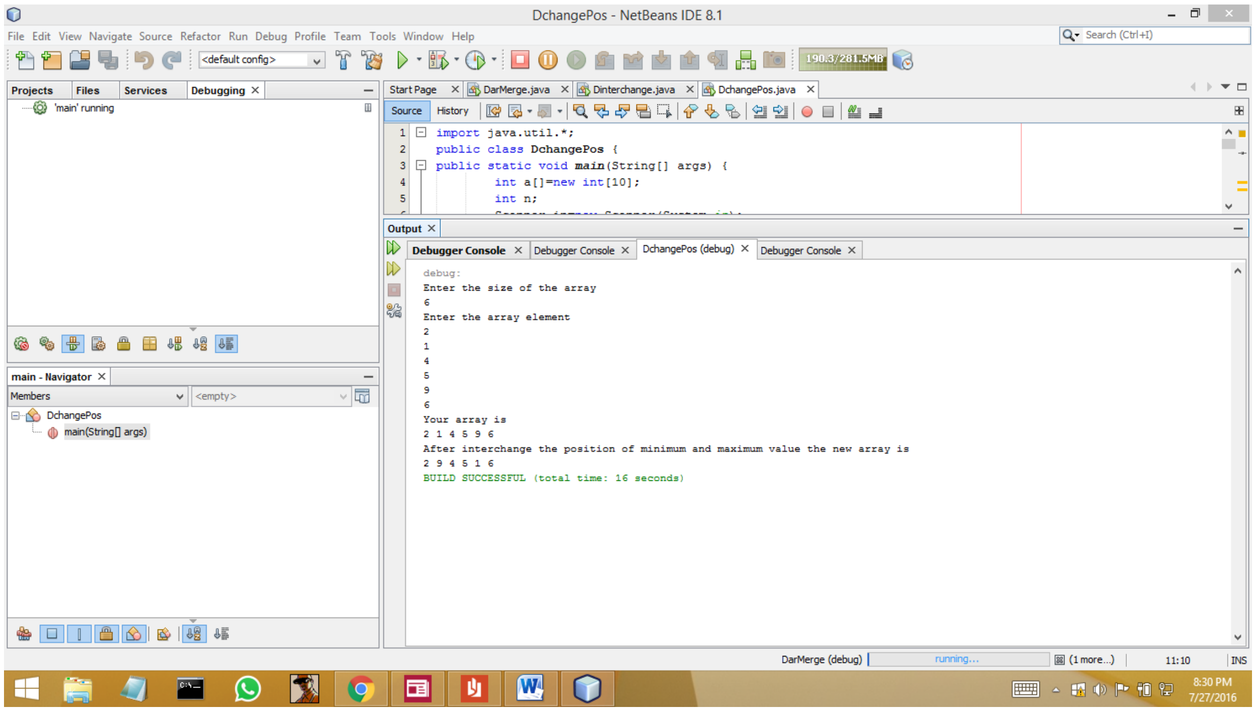
```
debug:
Enter the value of the n
8
Enter the Element of array
5
2
0
4
1
7
9
6
Position of 2nd smallest element is=5
BUILD SUCCESSFUL (total time: 10 seconds)
```

Dmin2pos (run) running... (6 more...) 46:1/46:927 INS

5. WAP to interchange largest and smallest element in an array.

```
import java.util.*;
public class DchangePos {
public static void main(String[] args) {
        int a[]=new int[10];
        int n;
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the size of the array");
        n=in.nextInt();
        System.out.println("Enter the array element");
        for(int i=0;i<n;i++)
        {
        a[i]=in.nextInt();
        }
        System.out.println("Your array is");
        for(int i=0;i<n;i++)
        {
        System.out.print(a[i]+" ");
        }
         System.out.println();
         int min=0;
         int max=1;
         for(int i=1;i<n;i++)
         {
         if(a[min]>a[i])
         {
         min=i;
         }
         }
         for(int i=0;i<n;i++)
         {
         if(i!=1)
         {
         if(a[max]<a[i])
         {
         max=i;
         }
         }
         }
         int temp=0;
         temp=a[min];
         a[min]=a[max];
         a[max]=temp;
         System.out.print("After interchange the position of minimum and ");
        System.out.print("maximum value the new array is");
        System.out.println();
        for(int i=0;i<n;i++)
            System.out.print(a[i]+" ");
            System.out.println();
        }
}
```

Output:



```
import java.util.*;
public class DchangePos {
public static void main(String[] args) {
        int a[]=new int[10];
        int n;
```

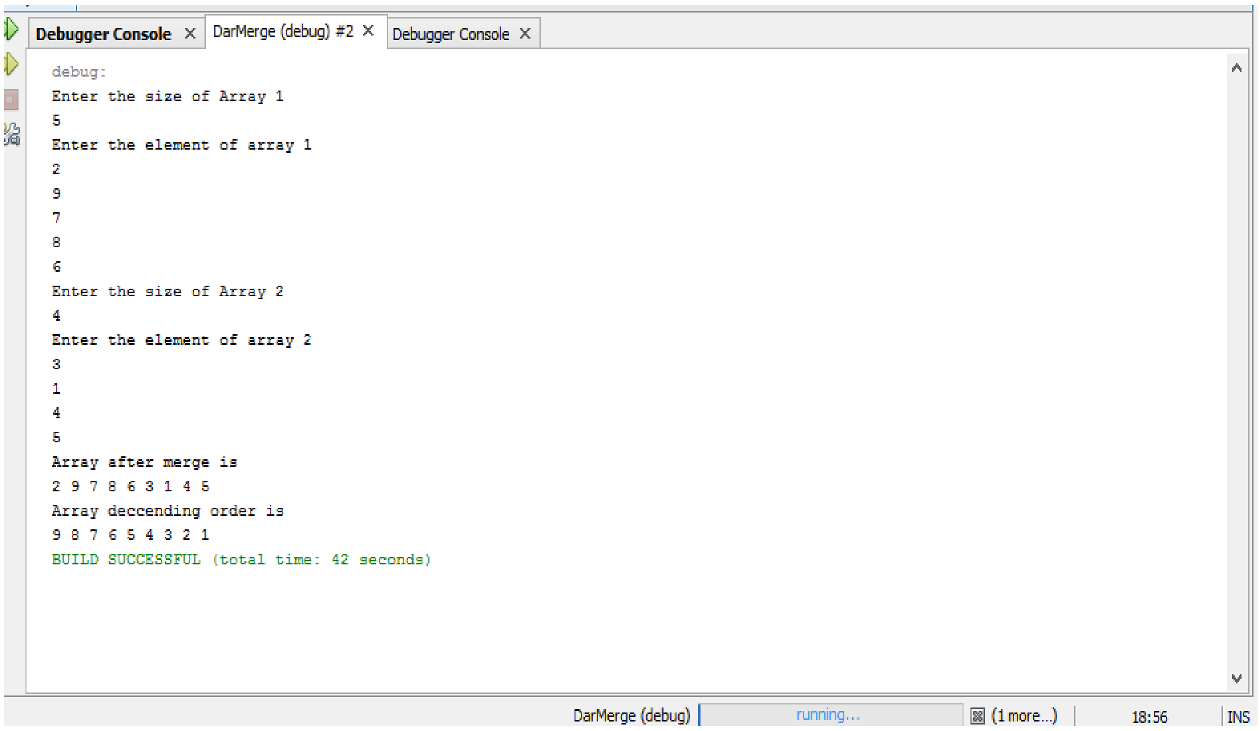
```
debug:
Enter the size of the array
6
Enter the array element
2
1
4
5
9
6
Your array is
2 1 4 5 9 6
After interchange the position of minimum and maximum value the new array is
2 9 4 5 1 6
BUILD SUCCESSFUL (total time: 16 seconds)
```

6.WAP to merge two array element in descending order:

```
import java.util.*;
public class DarMerge {
    public static void main(String[] args) {
      int a[]=new int[20];
      int b[]=new int[20];
      int c[]=new int[20];
      int flag=0;
      Scanner in=new Scanner(System.in);
      System.out.println("Enter the size of Array 1");
      int m=in.nextInt();
     System.out.println("Enter the element of array 1");
     for(int i=0;i<m;i++)
     {
     a[i]=in.nextInt();
     }
      System.out.println("Enter the size of Array 2");
      int n=in.nextInt();
      System.out.println("Enter the element of array 1");
     for(int i=0;i<n;i++)
     {
     b[i]=in.nextInt();
     }
   for(int i=0;i<m;i++)
   {
   c[i]=a[i];
   flag++;
   }
   for(int i=0;i<n;i++)
   {
   c[flag]=b[i];
   flag++;
```

```
    }
    System.out.println("Array after merge is");
    for(int i=0;i<m+n;i++)
    {
    System.out.print(c[i]+" ");
    }
    System.out.println("");
    for(int i=0;i<m+n-1;i++)
    {
    for(int j=i+1;j<m+n;j++)
    {
    if(c[i]<c[j])
    {
    int temp = c[i];
    c[i]=c[j];
    c[j]=temp;
    }
    }
    }
System.out.println("Array deccending order is ");
for(int i=0;i<m+n;i++)
{
System.out.print(c[i]+" ");
}
System.out.println();
}
}
```

Output:



```
debug:
Enter the size of Array 1
5
Enter the element of array 1
2
9
7
8
6
Enter the size of Array 2
4
Enter the element of array 2
3
1
4
5
Array after merge is
2 9 7 8 6 3 1 4 5
Array deccending order is
9 8 7 6 5 4 3 2 1
BUILD SUCCESSFUL (total time: 42 seconds)
```

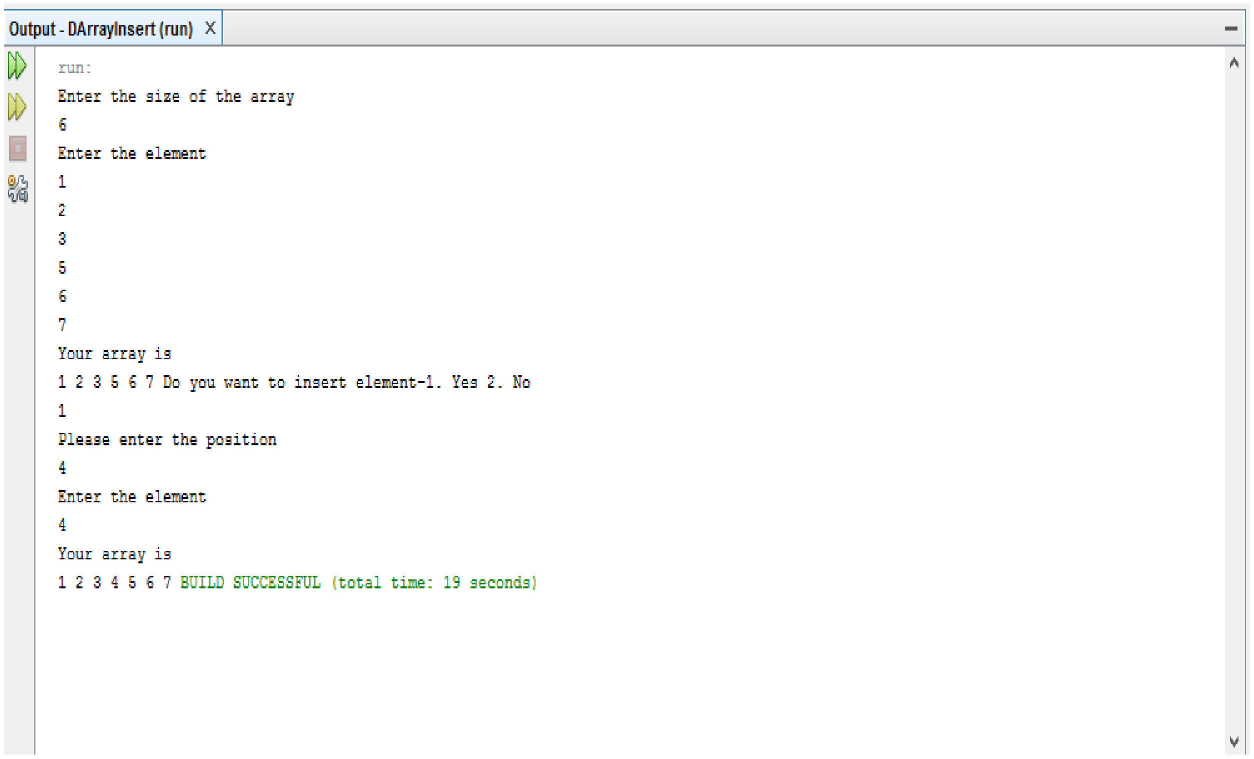
6. WAP to insert an element into array.

```
import java.util.*;
public class DArrayInsert {
   public static void main(String[] args) {
      int a[]=new int[20];
      Scanner in=new Scanner(System.in);
      System.out.println("Enter the size of the array");
      int n=in.nextInt();
      System.out.println("Enter the element");
      for(int i=0;i<n;i++)
      {
      a[i]=in.nextInt();
      }
      System.out.println("Your array is");
      for(int i=0;i<n;i++)
      {
```

```
System.out.print(a[i]+" ");
}
System.out.println("Do you want to insert element-1. Yes 2. No");
int ch=in.nextInt();
if(ch==1)
{
System.out.println("Please enter the position");
int p=in.nextInt();
if(p<=n)
{
for(int i=n-1;i>=p-1;i--)
{
a[i+1]=a[i];
}
System.out.println("Enter the element");
a[p-1]=in.nextInt();
System.out.println("Your array is");
for(int i=0;i<n+1;i++)
{
System.out.print(a[i]+" ");
}
}
else
{
System.out.println("Array position out of size");
}
}
else
```

```
        {
        System.out.println("See you again");
        }
      }
   }
```

Output:

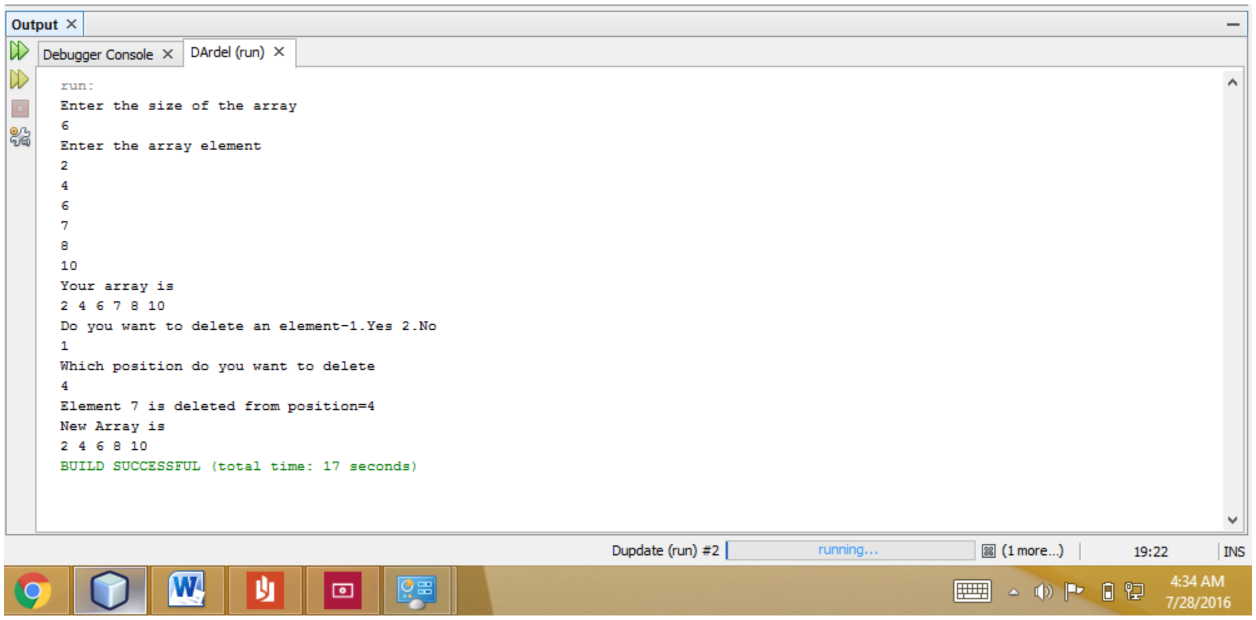


```
Output - DArrayInsert (run)
run:
Enter the size of the array
6
Enter the element
1
2
3
5
6
7
Your array is
1 2 3 5 6 7 Do you want to insert element-1. Yes 2. No
1
Please enter the position
4
Enter the element
4
Your array is
1 2 3 4 5 6 7 BUILD SUCCESSFUL (total time: 19 seconds)
```

7. WAP to delete a number from given position in an array.

```
import java.util.*;
public class DArdel {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the size of the array");
        int n=in.nextInt();
        int a[]=new int[n];
        System.out.println("Enter the array element");
        for(int i=0;i<n;i++)
        {
        a[i]=in.nextInt();
        }
        System.out.println("Your array is");
        for(int i=0;i<n;i++)
        {
        System.out.print(a[i]+" ");
        }
        System.out.println("");
        System.out.println("Do you want to delete an element-1.Yes 2.No");
        int ch=in.nextInt();
        if(ch==1)
        {
        System.out.println("Which position do you want to delete");
        int p=in.nextInt();
     System.out.println("Element "+a[p-1]+" is deleted from position="+p+"");
        if(p<=n)
        {
        for(int i=p-1;i<n-1;i++)
        {
        a[i]=a[i+1];
        }

        System.out.println("New Array is");
        for(int i=0;i<n-1;i++)
        {
        System.out.print(a[i]+" ");
        }
        System.out.println("");
        }
        else
        {
        System.out.println("Position is not found inside the array");
        }

        }
        else
        {
         System.out.println("Thank You");
        }
    }

}
```

Output:


```
run:
Enter the size of the array
6
Enter the array element
2
4
6
7
8
10
Your array is
2 4 6 7 8 10
Do you want to delete an element-1.Yes 2.No
1
Which position do you want to delete
4
Element 7 is deleted from position=4
New Array is
2 4 6 8 10
BUILD SUCCESSFUL (total time: 17 seconds)
```

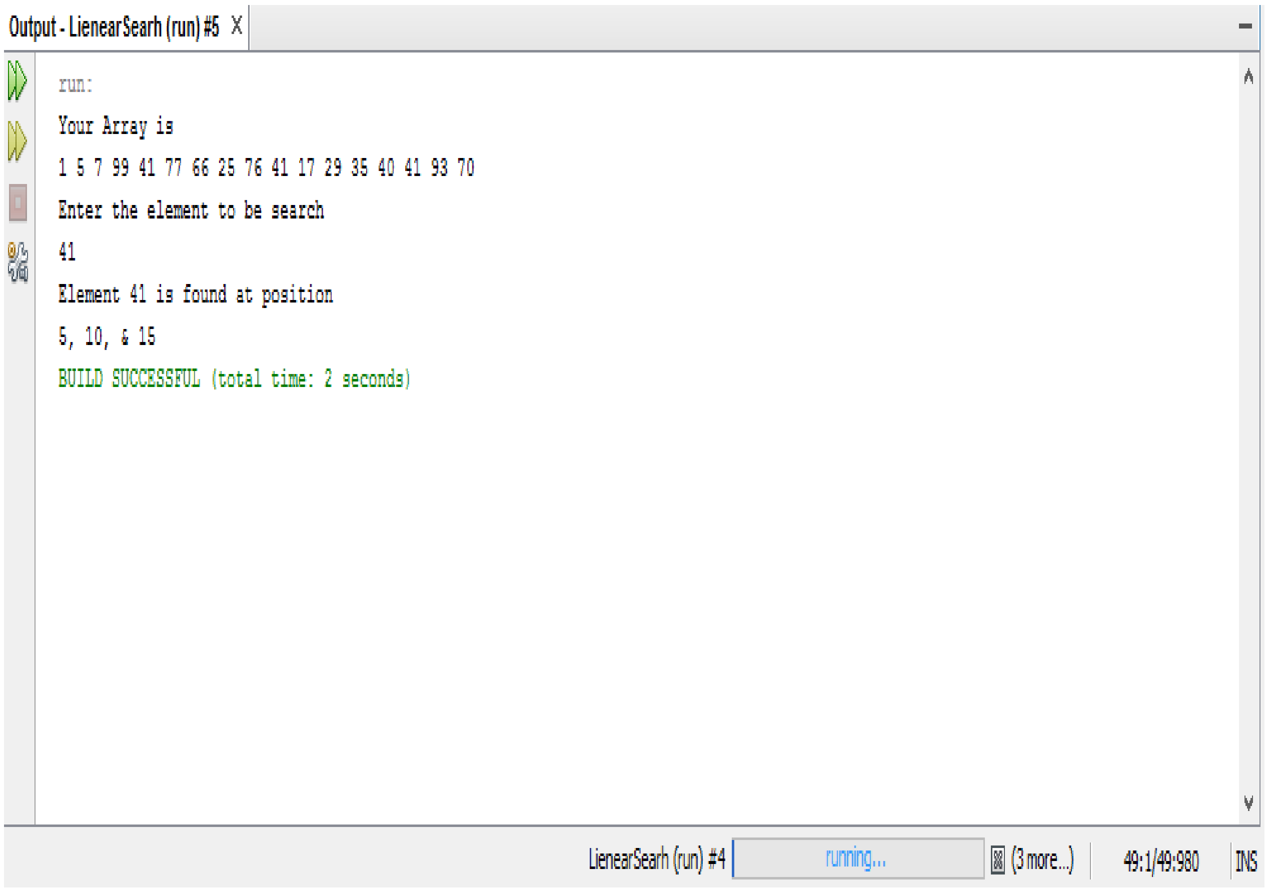
8. WAP to search an element from the list of elements using linear search technique

```
import java.util.*;
public class LienearSearh {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
    int a[]={1,5,7,99,41,77,66,25,76,41,17,29,35,40,41,93,70};
    System.out.println("Your Array is");
    for(int i=0;i<a.length;i++)
    {
    System.out.print(a[i]+" ");
    }
    System.out.println();
    int flag=0;
    int n=0;
    int ele[]=new int[a.length];
    System.out.println("Enter the element to be search");
    int s=in.nextInt();
```

```
for(int i=0;i<a.length;i++)
{
if(a[i]==s)
{   ele[n]=i+1;
    n++;
    flag++;
}
}
if(flag>=1)
{
System.out.println("Element "+s+" is found at position ");
for(int i=0;i<n;i++)
{
    if(i==n-1)
    {
    System.out.print("& "+ele[i]);
    }
else
    {
System.out.print(ele[i]+", ");
    }
}
    System.out.println();
}
else
{
System.out.println("Element is not found");
}
```

```
}
}
```

Output:



```
run:
Your Array is
1 5 7 99 41 77 66 25 76 41 17 29 35 40 41 93 70
Enter the element to be search
41
Element 41 is found at position
5, 10, & 15
BUILD SUCCESSFUL (total time: 2 seconds)
```