simple

```
public class FirstJavaProgram {
  public static void main(String[] args){
    System.out.println("This is my first program in java");
  }//End of main
}//End of FirstJavaProgram Class
```

```
This is my first program in java
```

Operators

```
public class ArithmeticOperatorDemo {
   public static void main(String args[]) {
      int num1 = 100;
      int num2 = 20;
      System.out.println("num1 + num2: " + (num1 + num2) );
      System.out.println("num1 - num2: " + (num1 - num2) );
      System.out.println("num1 num2: " + (num1 num2) );
      System.out.println("num1 / num2: " + (num1 / num2) );
      System.out.println("num1 % num2: " + (num1 % num2) );
      System.out.println("5.0 % 10.0: " + (5.0 % 10.0) );
      System.out.println("num1 + num2: " + num1 + num2 );
   }
}
```

```
num1 + num2: 120
num1 - num2: 80
num1 * num2: 2000
num1 / num2: 5
num1 % num2: 0

5.0 % 10.0: 0.0
```

the output is 0.5 so it gose to 0.0 even thoe it is a double

```
num1 + num2: 10020
```

Abstract

```
//abstract class
abstract class Sum{
   // Two abstract methods
   public abstract int sumOfTwo(int n1, int n2);
   public abstract int sumOfThree(int n1, int n2, int n3);

   //Regular method
   public void disp(){
System.out.println("Method of class Sum");
```

Abstract (cont)

```
   }
}
```

This code needs to be over rid by its child. it is a way to force the subclass to make certain methods.

super

```
//Parent class or Superclass or base class
class Superclass
{
   int num = 100;
}
//Child class or subclass or derived class
class Subclass extends Superclass
{
//The same variable num is declared in the Subclass
    int num = 110;
    void printNumber(){
System.out.println(num);
        System.out.println(super.num);
    }
        public static void main(String args[]){
Subclass obj= new Subclass();
obj.printNumber();
    }
}
```

```
110
100
```

Inheritance

```
Class X
{
   public void methodX()
   {
     System.out.println("Class X method");
   }
```

Inheritance (cont)

```
}
Class Y extends X
{
public void methodY()
{
System.out.println("class Y method");
}
}
Class Z extends Y
{
   public void methodZ()
   {
     System.out.println("class Z method");
   }
   public static void main(String args[])
   {
     Z obj = new Z();
     obj.methodX(); //calling grand parent class method
     obj.methodY(); //calling parent class method
     obj.methodZ(); //calling local method
  }
}
```

```
class X method
class Y method
class Z method
```

if else

```
public class IfStatementExample {
public static void main(String args[]){
      int num=70;
      if( num < 100 ) {
System.out.println("number is less than 100");
      }else {
           System.out.println("number is more less than 100");
      }
```

if else (cont)

```
   }
}
```

while

```
public class Test {
   public static void main(String args[]) {
      int x = 10;
      while( x < 20 ) {
         System.out.print("value of x : " + x );
         x++;
      }
   }
}
```

```
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
value of x : 15
value of x : 16
value of x : 17
value of x : 18
value of x : 19
```

do while

```
public class Test {
   public static void main(String args[]) {
      int x = 20;
      do {
         System.out.print("value of x : " + x );
         x++;
         System.out.print("\n");
      }while( x < 20 );
   }
}
```

```
value of x : 20
```

for

```
class ForLoopExample {
    public static void main(String args[]){
         for(int i=10; i>1; i--){
              System.out.println("The value of i is: "+i);
         }
    }
}
```

```
The value of i is: 10
The value of i is: 9
The value of i is: 8
The value of i is: 7
The value of i is: 6
The value of i is: 5
The value of i is: 4
The value of i is: 3
The value of i is: 2
```

swich

```
public class Test {
   public static void main(String args[]) {
      // char grade = args[0].charAt(0);
      char grade = 'C';
      switch(grade) {
         case 'A' :
            System.out.println("Excellent!");
            break;
         case 'B' :
         case 'C' :
            System.out.println("Well done");
            break;
         case 'D' :
            System.out.println("You passed");
         case 'F' :
            System.out.println("Better try again");
            break;
         default :
            System.out.println("Invalid grade");
      }
      System.out.println("Your grade is " + grade);
   }
```

swich (cont)

```
}
```

```
Well done
Your grade is C
```

Arrays

```
public class TestArray {
   public static void main(String[] args) {
      double[] myList = {1.9, 2.9, 3.4, 3.5};
      // Print all the array elements
      for (int i = 0; i < myList.length; i++) {
         System.out.println(myList[i] + " ");
      }

      // Summing all elements
      double total = 0;
      for (int i = 0; i < myList.length; i++) {
         total += myList[i];
      }
      System.out.println("Total is " + total);

      // Finding the largest element
      double max = myList[0];
      for (int i = 1; i < myList.length; i++) {
         if (myList[i] > max) max = myList[i];
      }
      System.out.println("Max is " + max);
   }
}
```

```
1.9
2.9
3.4
3.5
Total is 11.7
Max is 3.5
```

quizes

```
Condition 1: (x < y && x > 0) false
Condition 2: (a != d || x != 5) true
Condition 3: !(true && false) true
Condition 4: (x > y || a == 'A' || d != 'A') true
for x = 5, y = 3, and a and d are char variables with a = 'a' and d = 'A'
```

quizes (cont)

```
----------------------------------------------------------------------------------------------------
if (score >= 90)
     grade = 'A';
if (score >= 80)
     grade = 'B';
if (score >= 70)
     grade = 'C';
if (score >= 60)
     grade = 'D';
else
     grade = 'F';
//will work correctly only if score < 70, because it dosent brake. a A is also a C,D
----------------------------------------------------------------------------------------------------
if (x > 0)
     x++;
else
     if (x < 0)
          x--;
//adds one if x is pos and subs on if x is neg
----------------------------------------------------------------------------------------------------
if (count != 0 && total / count > max)
                 max = total / count;
//count = 0, The condition short circuits and the assignment statement is not executed nor dose it get to divitoin by 0
----------------------------------------------------------------------------------------------------
if (x < 0)
     y = x;
else
     y = 0;
// or
y = (x < 0) ? x : 0;
----------------------------------------------------------------------------------------------------
//Start: X = 1 end: x = 128, but if x = 0 it is an infinite lop
```

quizes (cont)

```
while (x < 100)
      x *= 2;
----------------------------------------------------------------------------------------------------
//will execute 10 times:
int x = 10;
while (x > 0)
{
     System.out.println(x);
     x--;
}
----------------------------------------------------------------------------------------------------
//end: x = 10: 1+2+3+4
for (int i=0; i<5; i++)
      x += i;
----------------------------------------------------------------------------------------------------
int x = 10;
do {
     System.out.println(x);
     x--;
} while (x > 0);
//executes 1 time
----------------------------------------------------------------------------------------------------
for (int j = 0; j < 1000; ) X++;
//is a infinite loop
----------------------------------------------------------------------------------------------------
for (int j = s.length( ); j > 0; j--)
     System.out.print(s.charAt(j-1));
//it prints s out backwards with the last character
----------------------------------------------------------------------------------------------------
int[ ] arr = new int[5];
arr.length = 5
int[ ] list = {5, 10, 15, 20, 6};
```

quizes (cont)

```
----------------------------------------------------------------------------------------------------
BankAccount[ ] firstEmpireBank;
firstEmpireBank = new BankAccount[1000];
//will creat 1,000 reference variables, each of which could point to a BankAccount object
----------------------------------------------------------------------------------------------------
```

variable

```
int numberOfDays;
 byte nextInStream;
 short hour;
 long totalNumberOfStars;
 float reactionTime;
 double itemPrice;
```

variable, Method and Class cunsructers

```
class XYZ{
   final void demo(){
      System.out.println("XYZ HI");
   }
}

class ABC extends XYZ{
   void demo(){
      System.out.println("ABC HI");
   } /* this method can not be crreated since the parnts method of the sasm name is final. howver we can just let it get inherited down.*/
   public static void main(String args[]){
      ABC obj= new ABC();
      obj.demo();
   }
}
------------------------------------
package abcpackage;
public class Addition {
   public int addTwoNumbers(int a, int b){
```

variable, Method and Class cunsructers (cont)

```
return a+b;
   }
}
package xyzpackage;
import abcpackage.*;
class Test{
   public static void main(String args[]){
      Addition obj = new Addition();
      System.out.println(obj.addTwoNumbers(100, 1));
   }
}
```

XYZ HI

101

garbage collection

```
BeginnersBook obj1 = new BeginnersBook();
BeginnersBook obj2 = new BeginnersBook();
obj2 = obj1;
char[] sayhello = { 'h', 'e', 'l', 'l', 'o'};
String str = new String(sayhello);
str = null;
```

str and obj2 *are now available for garbage collection.*
which means the instance (object) pointed by (referenced by) obj2 is not reachable

Encapsulation in Java

```
class EncapsulationDemo{
private String empName;
 public String getEmpName(){
        return empName;
    }
 public void setEmpName(String newValue){
        empName = newValue;
    }
}
```

Encapsulation in Java (cont)

```
public class EncapsTest{ // this class can not see empName
    public static void main(String args[]){
         EncapsulationDemo obj = new EncapsulationDemo();
         obj.setEmpName("Mario");
         System.out.println("Employee Name: " + obj.getEmpName());
    }
}
```

Employee Name: Mario
1) Make the instance variables private so that they cannot be accessed directlyfrom outside the class. You can only set and get values of these variables through the methods of the class.
2) Have getter and setter methods in the class to set and get the values of thefields.

DecimalFormat

```
DecimalFormat
void applyPattern(String pattern)
String format(Double number)
```

creates formatting object
applies a pattern to DecimalFormatobj
returns a string containing the number format according to current pattern

tax

```
//**
// Purchase.java Author: Lewis/Loftus
//
// Demonstrates the use of the NumberFormat class to format output.
//**
import java.util.Scanner;
import java.text.NumberFormat;
public class Purchase
{
   //-----------------------------------------------------------------
   // Calculates the final price of a purchased item using values
   // entered by the user.
   //-----------------------------------------------------------------
   public static void main (String[] args)
```

tax (cont)

```
   {
      final double TAX_RATE = 0.06; // 6% sales tax
      int quantity;
      double subtotal, tax, totalCost, unitPrice;
      Scanner scan = new Scanner (System.in);
      NumberFormat fmt1 = NumberFormat.getCurrencyInstance();
      NumberFormat fmt2 = NumberFormat.getPercentInstance();
      System.out.print ("Enter the quantity: ");
      quantity = scan.nextInt();
      System.out.print ("Enter the unit price: ");
      unitPrice = scan.nextDouble();
      subtotal = quantity * unitPrice;
      tax = subtotal * TAX_RATE;
      totalCost = subtotal + tax;
      // Print output with appropriate formatting
      System.out.println ("Total: " + fmt1.format(totalCost));
   }
}
```

enter quantity: 5
total: %20.51

ramdom

```
random()
float nextFloat()
int nextInt()
int nextInt(int num)
```

instagts random object
ranNum[0.0,0.1)
ranNum(all = or - n whole#)
ranNum(0,num-1)

enum

```
public enum Directions{
EAST,
WEST,
NORTH,
SOUTH
```

enum (cont)

```
}
 public static void main(String[] args) {
        EnumDemo obj1 = new EnumDemo(Directions.EAST);
        obj1.getMyDirection();
        Directions.charAt(2)
    }
}
```

the getMyDirection command is made to output the direction of the object

EAST
NORTH

String

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

//creating a string by java string literal
String str = "Beginnersbook";
char arrch[]={'h','e','l','l','o'};
//converting char array arrch[] to string str2
String str2 = new String(arrch);

//Displaying all the three strings
System.out.println(str);
System.out.println(str2);
   }
}
```

Beginnersbook
hello

constructor

```
class Example2
{
      private int var;
      //default constructor
      public Example2()
      {
             this.var = 10;
      }
```

constructor (cont)

```
      //parameterized constructor
      public Example2(int num)
      {
             this.var = num;
      }
public static void main(String args[])
      {
    Example2 obj = new Example2();
    Example2 obj2 = new Example2(100);
    System.out.println("var is: "+obj.getValue());
    System.out.println("var is: "+obj2.getValue());
      }
}
```

```
var is: 10
var is: 100
```

Static

```
class JavaExample{
  static int i = 100;
  static String s = "Beginnersbook";
  //Static method
  static void display()
  {
     System.out.println("i:"+i);
     System.out.println("i:"+s);
  }
  //non-static method
  void funcn()
  {
      //Static method called in non-static method
      display();
  }
  //static method
  public static void main(String args[])
```

Static (cont)

```
  {
JavaExample obj = new JavaExample();
//You need to have object to call this non-static method
obj.funcn();

      //Static method called in another static method
      display();
   }
}
```

i:100
i:Beginnersbook
i:100
i:Beginnersbook

intager

```
integer(int value)
prim typeValue()
// dyte byteValue() retuns number in primative data
static int parseInt(String str)
staic String toBinary{hex, Octal}String(int num)
```

creates new integer obj storing the value
returns number in primitive data
returns an int with value in the sting
returns string with the specified integer value in the corresponding base

format

```
String format(double number)
static NumberFormat getCurrencyInstamce()
static NumberFormat getPercenctInstamce()
//creates obj
// then do obj.format(Foo)
```

returns string with specified #format
returns object that represents current currency standers
returns object that represents current percentages standers