OBJECT ORINTED PROGRAMMING |

ASSIGNMENT – 01

1. Everything in Java is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has **attributes**, such as weight and color, and **methods**, such as drive and brake.
2. A Class is like an object constructor, or a "blueprint" for creating objects.
3. A **method** is a block of code which only runs when it is called. You can pass data, known as parameters, into a method. Methods are used to perform certain actions, and they are also known as **functions**.

1. The car has **attributes**, such as weight , color ,seat and tyres. Or you could say that class attributes are variables within a class:
2. 02. Compile time error.

03. a. 89

04. For example: in real life, a car is an object. The car has **attributes**, such as weight and color, and **methods**, such as drive and brake.

05. //Line 2

//Line 3

//Line 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

It’s supposed to be…..

b1.length

b1.width

b1.height

06. no value for length, width, height ;

07. volume : 180

Length of box : 12

Width of box : 5

Height of box : 3

08. default constructor

Length of box : 2

Width of box : 2

Height of box : 2

\*A constructor is called "Default Constructor" when it doesn't have any parameter.

09. parameterized constructor

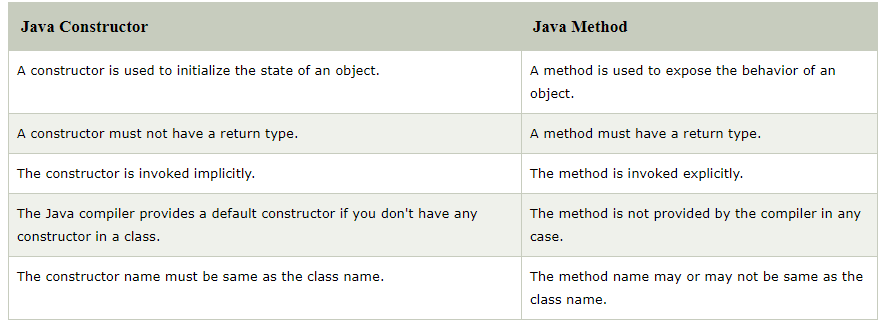
Volume : 60

Volume : 180

A constructor which has a specific number of parameters is called a parameterized constructor.

10.

11. The main purpose of a constructor is to initialize the **instance variables of a class**.

12. 

Every single person in this world has a unique name. Persons name helps us to identify them. So, name is the class name for every person. for example my name is

tharanga so tharanga is the name of my class.

**Class Khushboo{**

**}**

Now, we can say my behaviour and nature is mentioned in this class and this class represents me.

Now there is a situation when one person says my name, when he says “tharanga” I tried to figure out who is that person, I turned toward him and then asks him for what

reason he is calling me.

When that person says “tharanga”, he knowingly/unknowingly called default constructor tharanga.

This default constructor defines my response that I did (as mentioned above).

13.

A. Box b1;

B. Box b1=new Box();

c. new Box();

14. class box Not have attributes/properties like WIDTH and height

15. 1 2 3 4

16. A

B

D

17. 100 101

0 0

18. 3001

19. 2001

20. B. this is not constructor method. Just method.

C. default constructor method / not parmeterized.

D. compiler outomatically made constructor method ++++ A(){}

21.

\\LINE 01- create reference variable type Customer (variable name c1)

\\LINE 02- create an object type Customer

\\LINE 03-Methods of an object

\\LINE 04- create reference variable type Customer (variable name c2)

\\LINE 05- create an object type Customer

**`** \\ LINE 06-Encapsulation set method calling**(C2)**

\\LINE 07-calling method in object

\\LINE 08-create reference variable type Customer (variable name c3)

\\LINE 09- create an object type Customer

\\LINE 10- Encapsulation set method calling**(C3 SetId method)**

\\LINE 11- Encapsulation set method calling**(C3 SetName method)**

\\LINE 12- calling method in object(method name printcustomer)

22. A-

C-

D-

23. compiler error at line 2.

24. **Encapsulation in Java** is a process of wrapping code and data together into a single unit, for example, a capsule which is mixed of several medicines.

25.

Tight encapsulation is about ensuring that ALL data attributes are on the same objects as the relevant methods, and that only data attributes which need to be public are made public via getter/setter/properties - so to make the quad copter to start it’s engine - you have a method called - **start\_engine(engine\_speed),** and to hover you have a method : hover(height). So our code might look like :

1. **copter.start\_engine(1000) # Spin engines at 1000 rpm**
2. **....**
3. **copter.hover(500) # Hover at 500m**

Loose encapsulation is a less restrictive version, where Most key data attributes are on the same objects as their methods - but for one reason or another it isn’t possible to do that - it could be that you are extending legacy code, or that there is a performance issue. Another example of loose encapsulation would be to expose data attributes directly (as public attributes), rather than use getter/setters/properties which allow a level of control over the range of values, and behaviors when value are change. An example (for our quad copter) would be having to write code :

1. **copter.engine\_speed = 1000**
2. **copter.start()**
3. **....**
4. **copter.height = 500**
5. **copter.hover()**

26.

27. Compile Error al Line 1

Compile Error al Line 2

Compile Error al Line 3

28. E. 1 200 10 200 100 200

29. Compile Error Line-01

Compile Error Line-03

30. Compile Error al line – 05

Compile Error al line – 08

31. E. 1 200 10 200 100 200

32.

Instance Variable - A variable that is declared inside the class but outside the method is called instance variable . It is not declared as static.

Static Variable- A variable that is declared as static is called static variable. It cannot be local, It is not for any particulat object but for whole class

**class Test{**

**int iv=10;//instance variable**

**static int sv=100;//static variable**

**void method(){**

**int lv=50;//local variable**

**} }**

**33.  Instance method vs Static method**

* Instance method can access the instance methods and instance variables directly.
* Instance method can access static variables and static methods directly.
* Static methods can access the static variables and static methods directly.
* Static methods can’t access instance methods and instance variables directly. They must use reference to object. And static method can’t use [this](http://quiz.geeksforgeeks.org/this-reference-in-java/) keyword as there is no instance for ‘this’ to refer to.

34. Compile Error at line -01

Compile Error at line -04

Compile Error at line -06

Compile Error at line -09

35.

36.

package com.company;  
class rectangal{  
 private double length;  
 private double width;  
  
 rectangal(){  
 length=1;  
 width=1;  
 }  
 void setlength(double length){  
 this.length=length;  
 }  
 void setwidth(double width){  
 this.width=width;  
 }  
 double getlength(){  
 return this.length;  
 }  
  
 double getWidth(){  
 return this.length;  
 }  
  
  
 double perimetet(){  
 double perimetet;  
  
 if(length>0.0 && width > 0.0){  
 if(length<20.0 && width < 20.0){  
  
 perimetet = 2\*(length\*width);  
  
  
  
 return perimetet;  
 }  
 }  
 return -1 ;  
 }  
  
 double arae(){  
  
 double area;  
 if(length>0.0 && width > 0.0) {  
 if (length < 20.0 && width < 20.0) {  
 area=length\*width;  
  
 return area;  
 }  
 }  
  
 return -1;  
 }  
  
}  
class Demo {  
 public static void main(String args[]) {  
 rectangal r = new rectangal();  
  
 r.setlength(10);  
 r.setwidth(10);  
  
 double vol = r.getlength();  
 double nwx = r.getWidth();  
 System.out.println(vol + " " + nwx);  
  
 double perimetet = r.perimetet();  
 System.out.println("perimetet =" + perimetet);  
  
  
 double area = r.arae();  
 System.out.println("area =" + area);  
  
  
 }  
}

37. [ a , 0 ] , [ b , 1] , [ b , 1 ] , 1 , 0 , 0 , 2

38. prints : 1 2 3 4 5 3

48. (5 , 3) (3 , 5) (5 , 3)

49. Code 1001

Code 1001

50. package com.company;  
  
  
class box{  
  
 private int length;  
 private int width;  
 private int height;  
  
  
  
 box(int length){  
 this.length=length;  
 }  
  
 box(box b2) {  
 this.length=b2.length;  
 this.width=b2.width;  
 this.height=b2.height;  
 }  
  
  
 box(){};  
*//------------------------------------------------------------b2.--------------------------------------------------------* box(int length,int width,int height){  
 this.length=length;  
 this.width=width;  
 this.height=height;  
 }  
  
  
  
 *//------------------------------------------------------b2.b3.---------------------------------------------------------* void printvolume() {  
 int volume=length\*width\*height;  
 System.out.println("Volume "+volume);  
 }  
 *//------------------------------------------------------------------------------------------------------------------  
  
//--------------------------------------------------------b1.-----------------------------------------------------------* void setLength(int length){  
 this.length=length;  
 }  
  
 void setwidth(int width){  
 this.width=width;  
  
 }  
 void setheight(int height){  
 this.height=height;  
 }  
  
  
 public void setDimension(int length, int width, int height) {  
 this.length=length;  
 this.width=width;  
 this.height=height;  
  
 }  
  
 public void setDimensionb6(box b6) {  
 this.length=b6.length;  
 this.width=b6.width;  
 this.height=b6.height;  
  
 }  
 int getvolume() {  
 int volume = length \* width \* height;  
  
  
 return volume;  
 }  
*///----------------------------------------------------------b1.----------------------------------------------------------*}  
class Demo {  
 public static void main(String args[]) {  
  
 box b1=new box ();  
 b1.setLength(12);  
 b1.setwidth(5);  
 b1.setheight(3);  
 b1.setDimension(120,50,3);  
 System.out.println("volume "+b1.getvolume());  
  
 box b2 =new box (4,2,3);  
 b2.printvolume();  
  
 box b3=new box(b2);  
 b3.printvolume();  
  
  
 box b4=new box(10);  
 b4.printvolume();  
  
 box b5=new box();  
 b5.setDimension(2,0,0);  
 b5.printvolume();  
  
  
 box b6=new box();  
 *//b6.printvolume();* b6.setDimensionb6(b1);  
 b6.printvolume();  
  
  
 box b7=b3.getcopy();  
 b7.printvolume();  
 }  
}