[01]

1. Class/ Template - a set of objects which shares common characteristics/ behavior and common properties/ attributes
2. Object/ Instance - a basic unit of Object-Oriented Programming and represents real-life entities
3. Methods/ Functions - a block of code that, when called, performs specific actions mentioned in it
4. Attributes/ Properties - can be of any data type, including primitive types (like int , double , boolean ) and reference types (like String , arrays, or other objects
5. Reference Variables -  a variable that points to an object of a given class, letting you access the value of an object
6. Primitive Variables -  a primitive variable's information is stored as the value of that variable
7. Method Parameters -  a list of variables which tell us the type and order of variables that the method can accept
8. Local Variables - an auxiliary temporary variable that exists only while a particular function or a block of statements is executed

[2] D. Compile time error

[3] A. 89

[4]

[5] All are illegal

[6]

Volume : 0

length of box : 0

width of box : 0

height of box : 0

[7]

Volume : 180

length of box : 12

width of box : 5

height of box : 3

[8]

default constructor

length of box : 2

width of box : 2

height of box : 2

[9]

Parameterized constructor

Volume : 60

Parameterized constructor

Volume : 180

[10]

int length; - 0  
private int width; - error  
int height; - 0

[11] All are correct

[12]

 to create the instance of the class.

[13]

Constructor is used to create and initialize an Object. Method is used to execute certain statements

[14]

B. Box b1=new Box();

[15]

1 – 0

2 – error

3 – error

[16]

1 2 3 4

[17]

A. int x;

int y;

MyClass(int i, int j){ x=i; y=j; }

[18]

100 101

0 0

[19]

Code : 3001

[20]

B/C

[21]

A/B/D

[22]

Encapsulation in Java refers to integrating data (variables) and code (methods) into a single unit

[23]

When each variable is declared private in a particular class, it is commonly termed a “tightly encapsulated class”

[24]

D. Compiler error at line 2

[25]

//--------------------Date.java------------------------  
class Date{  
 int year=1970;  
 int month=1;  
 int day=1;  
  
 public int getYear() {  
 return year;  
 }  
  
 public void setYear(int year) {  
 this.year = year;  
 }  
  
 public int getMonth() {  
 return month;  
 }  
  
 public void setMonth(int month) {  
 this.month = month;  
 }  
  
 public int getDay() {  
 return day;  
 }  
  
 public void setDay(int day) {  
 this.day = day;  
 }  
   
 public void printDate(){  
 System.out.println();  
 }  
}  
  
//--------------------Demo.java------------------------  
class Demo{  
 public static void main(String args[]){  
 Date d1=new Date();  
 d1.printDate(); //1970-1-1  
 d1.year=2016; //Illegal  
 d1.month=5; //Illegal  
 d1.day=30; //Illegal  
 /\*year, month and day attributes  
 \*cannot be accessed to another class  
 \*/  
 d1.setYear(2016);  
 d1.setMonth(5);  
 d1.setDay(31);  
 System.out.println("Year : "+d1.getYear());  
 System.out.println("Month :"+d1.getMonth());  
 System.out.println("Day : "+d1.getDay());  
 }  
}

[27]

A. Compile Error at line 1

[28]

1 200 10 200 100 200

[29]

public class Q15 {  
 public static void main(String[] args) {  
  
 Rectangle r1=new Rectangle(10,15);  
 r1.calcArea();  
 }  
}  
  
class Rectangle{  
  
 private double length=1.0;  
 private double width=2;  
  
 Rectangle(double length, double width){  
 this.length=length;  
 this.width=width;  
 }  
  
 public double getLength() {  
 return length;  
 }  
  
 public void setLength(double length) {  
 this.length = length;  
 }  
  
 public double getWidth() {  
 return width;  
 }  
  
 public void setWidth(double width) {  
 this.width = width;  
 }  
  
 public void calcArea(){  
 if (length>0.0 && width<20.0){  
 double area=width\*length;  
  
 System.out.println(area);  
 }  
 }  
}

[30]

A. Compile Error at line 1

[31]

1 200 10 200 100 200

[32]

Line ¼

[33]

A static variable is associated with the class itself rather than with any specific instance of the class. In contrast, an instance variable is associated with a specific instance of a class, and each instance has its own copy of that variable.

[34]

Line 1/4/5/6/7/8/9

[35]

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

[36]

C

[40]

Box is loaded into memory

[41]

A box object is created..

A box object is created..

[42]

Box is loaded into memory

A box object is created..

A box object is created..

A box object is created..

[43]

C/E/F

[44]

C/D

[45]

Constructor overloading allows a class to have multiple constructors with different parameter lists

[46]

C/G/K/M/D/F/H/J/N

[47]