1 a) (d)String[]

b) (c)false,0

c) (a)break

d) (c)return

2 a) import java.util.Scanner;

class CheckEvenOdd

{

public static void main(String args[])

{

int num;

System.out.println("Enter an Integer number:");

Scanner input = new Scanner(System.in);

num = input.nextInt();

if ( num % 2 == 0 )

System.out.println("Entered number is even");

else

System.out.println("Entered number is odd");

}

}

Enter an Integer number:

34

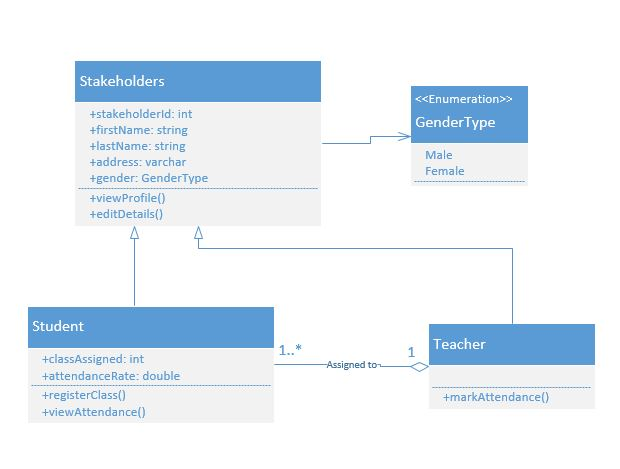
Entered number is even

OUTPUT

b)

1,2,4,5,

3 a)



b)There are four types of Java access modifiers:

1. **Private**: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
2. **Default**: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
3. **Protected**: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
4. **Public**: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.
5. Let's understand the access modifiers in Java by a simple table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Access Modifier** | **within class** | **within package** | **outside package by subclass only** | **outside package** |
| **Private** | Y | N | N | N |
| **Default** | Y | Y | N | N |
| **Protected** | Y | Y | Y | N |
| **Public** | Y | Y | Y | Y |

c) In [Java](https://www.javatpoint.com/java-tutorial), a constructor is a block of codes similar to the method. It is called when an instance of the [class](https://www.javatpoint.com/object-and-class-in-java) is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object.

There are two rules defined for the constructor.

1. Constructor name must be the same as its class name
2. A Constructor must have no explicit return type
3. A Java constructor cannot be abstract, static, final, and synchronized

Every time an object is created using the new() keyword, at least one constructor is called.

There are two types of constructors in Java:

1. Default constructor (no-arg constructor) : A constructor is called "Default Constructor" when it doesn't have any parameter.

|  |
| --- |
| It will be invoked at the time of object creation. |

1. //Java Program to create and call a default constructor
2. **class** Bike1{
3. //creating a default constructor
4. Bike1(){System.out.println("Bike is created");}
5. //main method
6. **public** **static** **void** main(String args[]){
7. //calling a default constructor
8. Bike1 b=**new** Bike1();
9. }
10. }

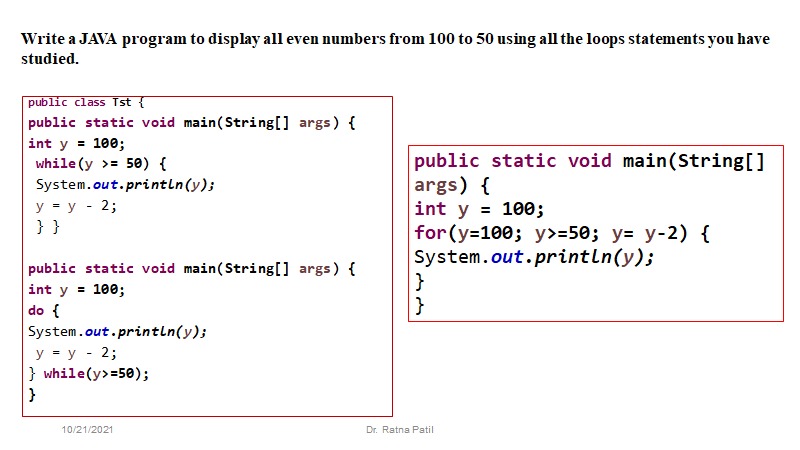
2 Paramertized constructor : A constructor which has a specific number of parameters is called a parameterized constructor.

The parameterized constructor is used to provide different values to distinct objects. However, you can provide the same values also.

1. //Java Program to demonstrate the use of the parameterized constructor.
2. **class** Student4{
3. **int** id;
4. String name;
5. //creating a parameterized constructor
6. Student4(**int** i,String n){
7. id = i;
8. name = n;
9. }
10. //method to display the values
11. **void** display(){System.out.println(id+" "+name);}
13. **public** **static** **void** main(String args[]){
14. //creating objects and passing values
15. Student4 s1 = **new** Student4(111,"Karan");
16. Student4 s2 = **new** Student4(222,"Aryan");
17. //calling method to display the values of object
18. s1.display();
19. s2.display();
20. }

4 a) The four pillars for OOP are Abstraction, Encapsulation, Inheritance, Polymorphism.

1. Abstraction : Abstraction is the process of showing only essential/necessary features of an entity/object to the outside world and hide the other irrelevant information. For example to open your TV we only have a power button, It is not required to understand how infra-red waves are getting generated in TV remote control.
2. Encapsulation : Encapsulation means wrapping up data and member function (Method) together into a single unit i.e. class. Encapsulation automatically achieve the concept of data hiding providing security to data by making the variable as private and expose the property to access the private data which would be public.
3. Inheritance : The ability of creating a new class from an existing class. Inheritance is when an object acquires the property of another object. Inheritance allows a class (subclass) to acquire the properties and behavior of another class (super-class). It helps to reuse, customize and enhance the existing code. So it helps to write a code accurately and reduce the development time.
4. Polymorphism: Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means "many forms". A subclass can define its own unique behavior and still share the same functionalities or behavior of its parent/base class. A subclass can have their own behavior and share some of its behavior from its parent class not the other way around. A parent class cannot have the behavior of its subclass.

b)

5 a) public class Main {

public static void main(String[] args) {

String name = args[0];

String dep = args[1]

int sub1 = Integer.parseInt(args[2]);

int sub2 = Integer.parseInt(args[3]);

int sub3 = Integer.parseInt(args[4]);

int sub4 = Integer.parseInt(args[5]);

int totalMarks = sub1+sub2+sub3+sub4;

int averageMarks = totalMarks/4;

System.out.println("Name is "+ name) ;

System.out.println("Department is"+ dep) ;

System.out.println("Total marks of student is: "+totalMarks+"\n The average mark of student is: "+averageMarks);

}

}

Name is Akansha

Department is IT

98

85

90

Total marks of student is: 273

The average mark of student is: 91

b)

1. import java.util.Scanner;
2. class ReverseNumber {
3. public static void main(String args[]) {
4. int n, reverse = 0;
5. System.out.println("Enter the number to reverse");
6. Scanner in = new Scanner(System.in);
7. n = in.nextInt();
8. while( n != 0 ) {
9. reverse = reverse \* 10;
10. reverse = reverse + n%10;
11. n = n/10;
12. }
13. System.out.println("Reverse of entered number
14. is “+reverse);
15. }
16. }

Enter the number to reverse

245

Reverse of entered number is 542