```
class Test{

	public static void add(){
	int a,b,c;
	a=10;
	b=20;
	c=a+b;
	System.out.println("Addition : "+c);

	}

public static void add(int a){
	int c;
	c=a+a;
	System.out.println("Addition : "+c);

	}

public static void add(int a,int b){
	int c;
	c=a+b;
	System.out.println("Addition : "+c);
```

```
    }

public static void add(int a,int b,int d){
    int c;
    c=a+b+d;
    System.out.println("Addition : "+c);

    }
public static void add(int a,int b,int d,int
e){
    int c;
    c=a+b+d+e;
    System.out.println("Addition : "+c);

    }
    public static void add(int f,int a,int
b,int d,int e){
    int c;
    c=a+b+d+e+f;
    System.out.println("Addition : "+c);

    }
```

```
	public static void add(int g,int f,int
a,int b,int d,int e){
	int c;
	c=a+b+d+e+f+g;
	System.out.println("Addition : "+c);

	}

	public static void main(String…args){

	add();
	add(10);
	add(1,2);
	add(10,20,30);
	add(10,20,30,40);
	add(1,2,3,4,5);
	add(1,2,3,4,5,6);
	}


}
```

Q1. Explain Variable Argument(…) in Java Programming?

Ans : In Java Variable arguments(varargs) allow a method to accept zero or more arguments of the same type. This is useful when exact number of arguments is unknown or varies.

Here datatype… indicates the method can accept a variable number of arguments of the specified data types

Example:

```
void add(int…a){
}
void add(float…a){
}

void add(double…a){
}
void add(String…a){
}
```

Example : Variable Argument

```
class Test{

    public static void add(int...x){
    int sum=0;
    for(int a:x){
    sum=sum+a;
    }
    System.out.println("Sum is : "+sum);
    }

    public static void main(String...args){

    add();
    add(10);
    add(1,2);
    add(10,20,30);
    add(10,20,30,40);
    add(1,2,3,4,5);
    add(1,2,3,4,5,6);
    }
```

}

Note:

1. The varargs parameter is treated as an array within the method.
2. A Method can have exactly one varargs parameter
3. The varargs parameter must be the last parameter in the method

```
class Test{

    public static void add(int y,int...x){
    int sum=0;
    for(int a:x){
    sum=sum+a;
    }
    System.out.println("Sum is : "+sum);
    }

    public static void main(String...args){

    //add();
    add(10);
```

```
    add(1,2);
    add(10,20,30);
    add(10,20,30,40);
    add(1,2,3,4,5);
    add(1,2,3,4,5,6);
    }


}
```

Q2. Explain Abstraction in java Programming?

Ans: Abstraction in java is a process to hiding the implementation details of a class and showing only the essential features. It is achieved by abstract class and interface. Abstraction helps to reduce complexity and increase maintainability

An abstract class is a class that is declared with the abstract keyword. Abstract class can contain abstract method or non

abstract method. An Abstract class can have constructor

We cannot create an instance of abstract class But it can store reference of its child class

In abstract class we can provide only declaration the method we cannot implement it inside abstract class.

If we extends any abstract class we must override its abstract method or we declare child class also abstract class.

College - Notepad

File Edit Format View Help

```
abstract class Rgpv{
        public abstract void lab();
        public abstract void cctv();
        public void copyCheck(){
        System.out.println("This is Copy Check Method of RGPV class and this is non
abstract method");
        }

}

abstract class College extends Rgpv{

}
```

Activate Windows
Go to Settings to activate Windows.

Ln 12, Col 2 | 60% | Windows (CRLF) | UTF-8

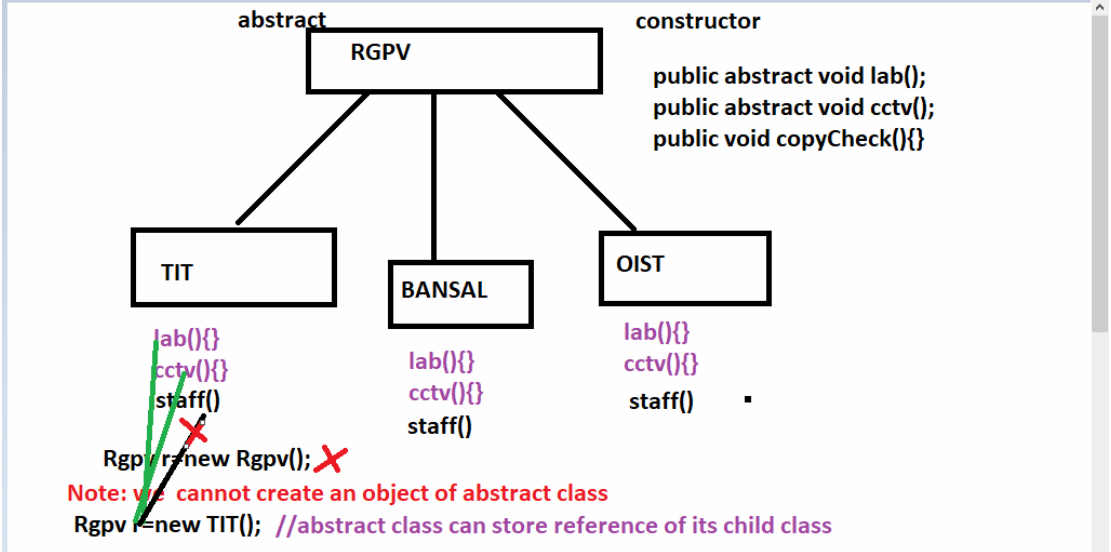
Type here to search | 16°C | ENG | 20:18 08-01-2025

Untitled - Paint

File Home View

Paste | Cut | Copy | Crop | Resize | Rotate | Select | Brushes | Outline | Fill | Size | Color 1 | Color 2 | Edit colors | Edit with Paint 3D

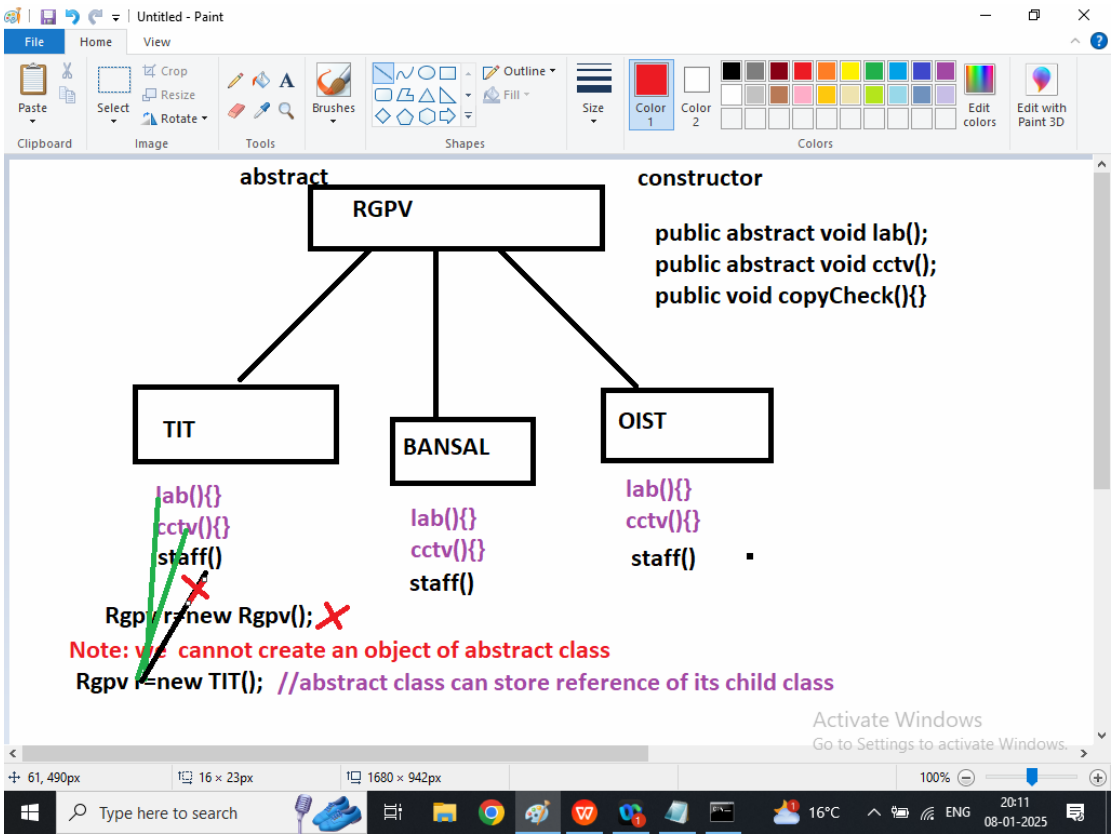
Clipboard | Image | Tools | Shapes | Colors



Activate Windows
Go to Settings to activate Windows.

16 × 23px | 1680 × 942px | 100%

Type here to search | 16°C | ENG | 20:14 08-01-2025



```
abstract class Rgpv{
    public Rgpv(){
    System.out.println("This is RGPV Class Constructor");
    }
    public abstract void lab();
    public abstract void cctv();
    public void copyCheck(){
    System.out.println("This is Copy Check Method of RGPV class and this is non abstract method");
    }
```

```
}

class College extends Rgpv{
public College(){
      System.out.println("This is College
Class Constructor");
      }
public void cctv(){
System.out.println("This is Rgpv CCTV
method");
}
public void lab(){
System.out.println("This is Rgpv lab
method");
}
public void staff(){
System.out.println("This is College staff
method");
}

public static void main(String args[]){
Rgpv r=new College();
```

```
r.cctv();
r.lab();
r.copyCheck();
//r.staff();
College t1=new College();
t1.cctv();
t1.lab();
t1.copyCheck();
t1.staff();
}
}
```