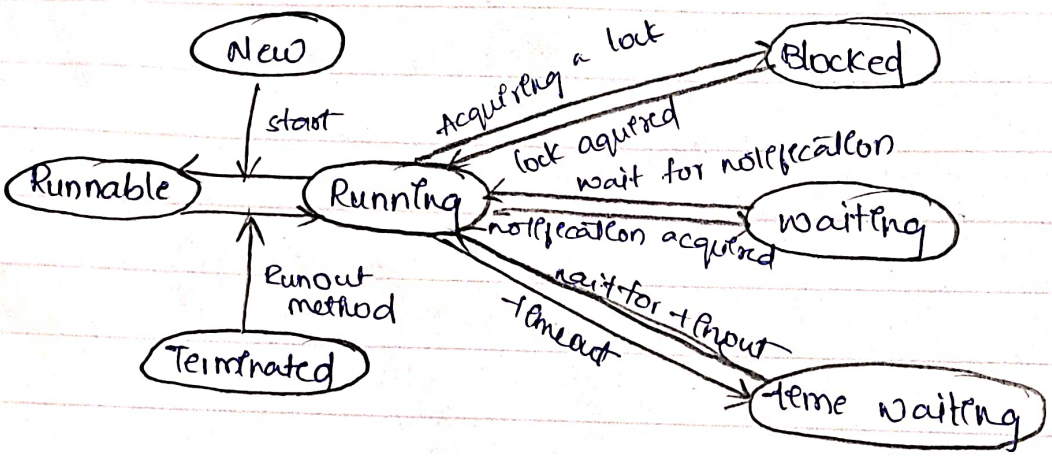
① Define Thread. Explain Thread life cycle with suitable program.

Thread allows a program to operate more efficiently by doing multiple things at the same time. Thread can be used to perform complicated tasks in the background without interrupted the main program.

Thread Life cycle:



New Thread:

Thread state for a thread that has not yet started.

Runnable:

Thread state for a runnable thread. But it may waiting for other resources.

Blocked:

Thread state for a thread blocked waiting for a monitor lock.

Waiting:

The thread is in the waiting state due to calling one of the following methods: object.wait with no timeout.

Thread.join with no timeout.

Terminated:
Thread state for a terminated thread. The thread has completed execution.

Example:

```
class threadExample implements Thread
{ public void run()
  { try { Thread.sleep(1000); }
    catch (InterruptedException e)
    { e.printStackTrace(); }
  System.out.println("state of thread1 while it called join() method on
  thread2 -" + Test.thread1.getState());
  try { Thread.sleep(1000); }
  catch (InterruptedException e)
  { e.printStackTrace(); } }
public class Test implements Thread
{ public static Thread thread1;
  public static Test obj;
  public static void main (String[] args)
  { obj = new Test();
    thread1 = new Thread(obj);
    System.out.println("State of thread1 after creating it-" + thread1.get
    State());
    thread1.start();
    system.out.println("state of thread1 after calling .start() method on
    it-" thread1.getState());
}
```

```
    public void run()
{ threadExample mythread = new threadExample();
    Thread thread2 = new Thread(mythread);
    System.out.println("state of thread2 after creating it -"+thread2.get
    state());
  try { Thread.sleep(1000); }
  catch (InterruptedException e) { e.printStackTrace(); }
  System.out.println("state of thread2 after calling .sleep() method on
  it - " + thread2.getState());
try { thread2.join(); }
catch(InterruptedException e)
  { e.printStackTrace();
  }
  System.out.println("state of thread2 when it has finished its
execution - "+thread2.getstate());
 }
}
```

output:

State of thread1 after creating it - ~~NEW~~ NEW

State of thread1 after calling .start() method on it - RUNNABLE

State of thread2 after creating it - NEW

State of thread2 after calling .start() method on it - RUNNABLE

State of ~~thread2~~ thread2 while it called

State of thread2 after calling .sleep() method on it - TIMED_WAITING

State of thread1 while it called join() method on thread2 - WAITING

State of thread2 when it has finished its execution - TERMINATED.

(2) Write a java program to handle producer - consumer problem using inter thread communication mechanism.

```
package package1;
class Q
{ int n
  boolean valueset = false;
 synchronized public void get()
  { if (!valueset)
    { try { wait(); }
      catch (Exception e)
       { system.out.println(e)}
    }
   System.out.println("Get" +n);
   try { Thread.sleep(1000); }
   catch (Exception e) { }
   valueset=false;
   notify();
}
Synchronized public void put (int nn)
{ if (valueset)
  { try { wait(); }
    catch (exception e)
      { system.out.println(e) }
  }
  n=nn;
  System.out.println ("put" +n);
```

```
        try{ Thread.sleep(1000);}
        catch(exception e)
        { system.out.println(e);}.
        valueset= true;
        notify();
    }}
class producer extends Thread
{ Q qq
    public producer (Q qq)
    { q=qq;
    }
    @override
    public void run()
    { int i=0;
      while (true)
    { q.put(i++);
    }
  }
}
class consumer extends Thread
 { Q q;
    public consumer(Q qq)
    { q=qq;
    }
  @override
    public void run()
{ int i=0;
    while (true);
    { q.get();
    }
  }
}
```

```
public class Test
{ public static void main (String[] args)
  { Q q = new Q();
    Producer p = new Producer(q);
    Consumer c = new Consumer(q);
    p.start();
    c.start();
  }
}
```

Output:

```
put:0
get:0
put:1
get:1
```

③ Explain Generics in java with suitable programs?

Generics in java allows you to create classes, Interfaces and methods with type parameters, enabling you to write code that can work with different data types.

The object is the superclass of all other classes, and object reference can refer to any object.

Example:

```
class Test <T, U>
{ T obj1;
  U obj2;
  Test (T obj1, U obj2)
  { this.obj1 = obj1;
    this.obj2 = obj2;
  }
```

```
public void print()
 { System.out.println(obj1);
  System.out.println(obj2);
 }
}

class main()
 { public static void main(String[] args)
  { Test<String, Integer> obj = new Test<String, Integer>
                                ("Geneecs", 1);
    obj.print();
  }
}
```

output:

Geneecs
1

④ Explain any two different layout managers with example program.

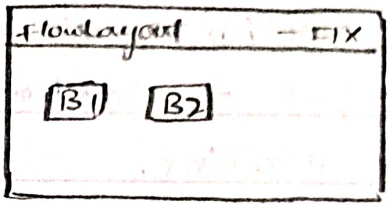
FlowLayout:

FlowLayout arranges components in a left-to-right, top-to-bottom

Ex:
```
import javax.swing.*;
import java.awt.*;
public class example{
public static void main(String[] args)
 { JFrame f = new JFrame("flowlayout");
   f.setSize(300,200);
   f.setDefaultCloseOperation(JFrame.EXITONCLOSE);
   JPanel p = new JPanel(new flowlayout());
   p.add(new JButton("B1");
```

```
p.add(new JButton("B2"));
f.add(p);
f.setVisible(true);
}}
```

output



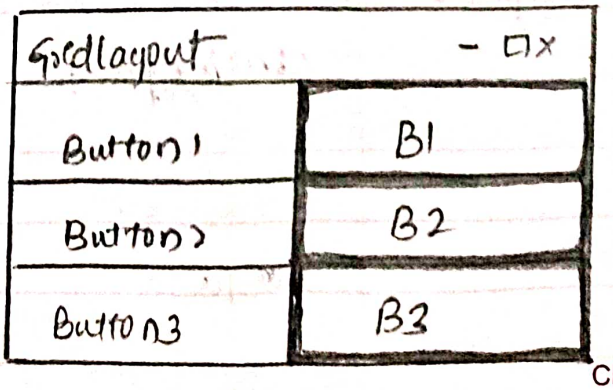
GridLayout:

GridLayout organizes components in a grid format, where each cell in the grid holds one component.

Ex:

```
import javax.swing.*;
import java.awt.*;
public class example
{ public static void main(String[] args)
 { JFrame f = new JFrame("Grid Layout");
f.setSize(300,200);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 JPanel p = new JPanel(new GridLayout(3,2));
  p.add(new JLabel("Button 1"));
  p.add(new JButton("B1"));
 p.add(new JLabel("Button 2"));
 p.add(new JButton("B2"));
 p.add(new JLabel("Button 3"));
 p.add(new JButton("B3"));
 f.add(p);
f.setVisible(true);
}}
```

output:

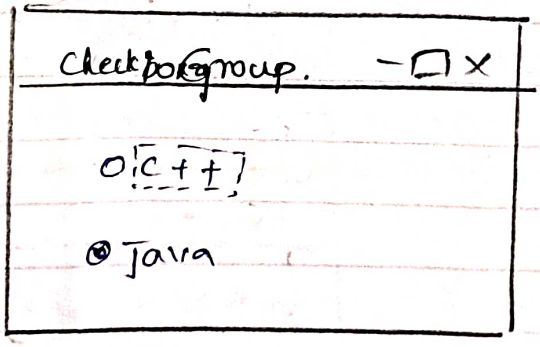


(6)(8) a) Explain checkbox groups and choices of AWT control in Java.

The object of checkboxgroup class is used to group together a set of checkbox.

Ex:
```
import java.awt.*;
public class checkboxgroupex
{ checkboxgroupex()
  { frame f = new frame("checkbox group");
    CheckboxGroup c = new checkboxGroup();
    Checkbox c1 = new Checkbox("C++", c, false);
    c1.setBounds(100,100,50,50);
    Checkbox c2 = new Checkbox("java", c, true);
    c2.setBounds(100,150,50,50);
    f.add(c1); f.add(c2);
    f.setSize(300,200);
    f.setLayout(null);
    f.setVisible(true);
  }
  public static void main(String[] args)
  { new checkboxgroupex();
  }
}
```

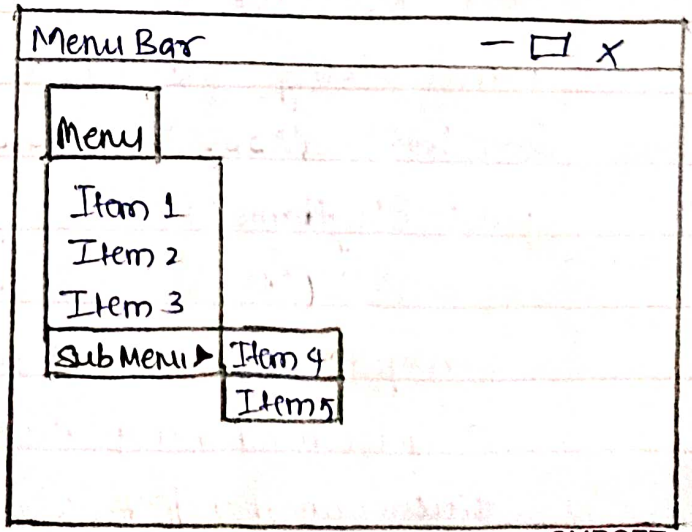
Output:



b) Write a java program to develop menubar.

```
import javax.swing.*;
class Example
{  JMenu  menu, submenu;
   JMenuItem  m1, m2, m3, m4, m5;
   Example()
   {  JFrame f = new JFrame("MenuBar");
      JMenuBar m = new JMenuBar();
      menu = new JMenu("Menu");
      submenu = new JMenu("subMenu");
      m1 = new JMenuItem("Item 1");
      m2 = new JMenuItem("Item2");
      m3 = new JMenuItem("Item 3");
      m4 = new JMenuItem("Item 4");
      m5 = new JMenuItem("Item 5");
      menu.add(m1); menu.add(m2); menu.add(m3);
      submenu.add(m4); submenu.add(m5);
      menu.add(submenu);
      m.add(menu);
      f.setJMenuBar(m);
      f.setSize(400,400);
      f.setLayout(null);
      f.setVisible(true);
   }
   public static void main(String[] args)
   {  new Example();
   }
}
```

output :



CMR GROUP OF INSTITUTIONS
EXPLORE TO INVENT

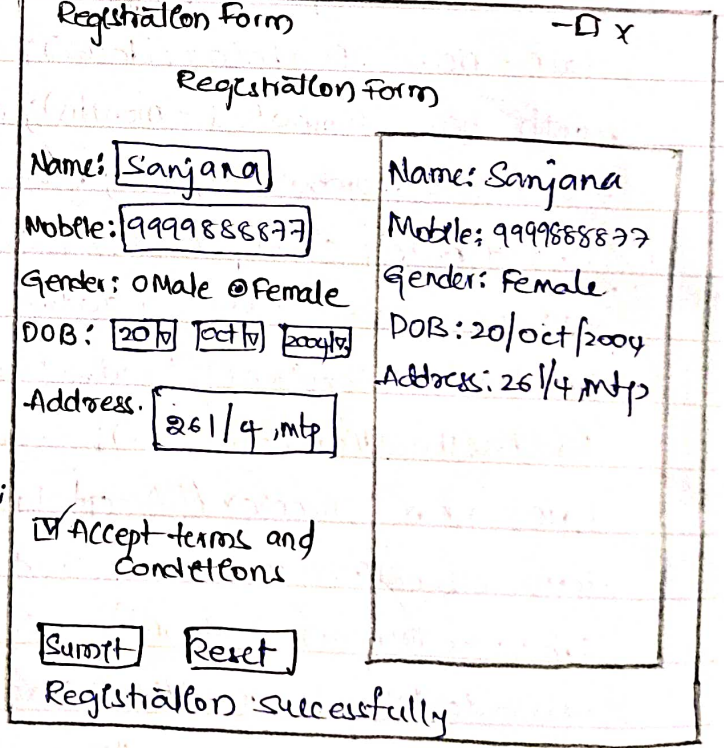
(5) Write a swing program to demonstrate JOB Registration form with the following data i) Name ii) password iii) email iv) Contactno. v) gender vi) languages known vii) city, after submit button, display message as "Registration successful".

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class Myframe extends JFrame implements ActionListener
{ private Container c;
  private JTextField tname, tmno; tadd,
  private JLabel title, name, mno, gender, dob, add, res;
  private JRadioButton male, female;
  private JComboBox date, month, year;
  private ButtonGroup gl;
  private JTextField JTextArea tadd, tout, resadd;
  private JButton sub, reset;
  private JCheckBox term;
 private string dates[] = {"1","2","3","4","5","6","7","8","9","10","11","12",
"13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};
private string months[] = {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep",
"Oct","Nov","Dec"};
private string years[] = {"2000","2000","2002","2003","2004","2005","2006","2007
"2009","2010","2011","2012","2013","2014","2015","2016","2017","2018","2019","2020","2021","2022","2023",
"2024"};
public Myframe()
{ setTitle("Registration Form"); setSize(300,200);
  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  c = getContentPane(); c.setLayout(null);
  title = new JLabel("Registration form"); title.setSize(300,30);
  title.setLocation(300,30); c.add(title);
```

```
name = new JLabel("Name"); name.setSize(100, 20); name.setLocation(100,100);
c.add(name);
tname = new JTextField(); tname.setSize (190,20);
tname.setLocation (200,100); c.add(tname);
mno = new JLabel ("Mobile"); mno.setSize (100,20);
mno.setLocation (100,150); c.add(mno);
gender = new JLabel ("Gender"); gender.setSize(100,20); gender.setLocation(100,200);
c.add(gender);
male = new JRadioButton ("Male"); male.setSelected(true); male.setSize(50,20);
male.setLocation(200,200); c.add(male);
female=new JRadioButton ("female"); female.setSelected(true); female.setSize(70,20);
female.setLocation (275,200); c.add (female);
g1 = new ButtonGroup(); g1.add(male); g1.add(female);
date = new JComboBox (dates); date.setSize (50,20); date.setLocation (200;250);
month = new JComboBox (months); month.setSize(50,20); month.setLocation(250,250);
year = new JComboBox(years); year.setSize(50,20); year.setLocation (320,250);
c.add(date); c.add(month); c.add (year);
add = new JLabel ("Address"); add.setSize(100,20); add.setLocation (100, 300);
tadd = new JTextArea(); tadd.setSize(200,25); tadd.setLineWrap(true);
tadd.setLocation (200,300); c.add(add); c.add(tadd);
term = new JCheckBox ("Accept terms & conditions"); term.setSize (250,20);
term.setLocation (150,400); c.add (term);
sub = new JButton ("submit"); sub.setSize(100,20); sub.setLocation (150, 450);
sub.addActionListener(this); c.add(sub);
reset = new JButton ("Reset"); reset.setSize (100,20); reset.setLocation (270,450);
reset.addActionListener(this); c.add(reset);
tout = new JTextArea(); tout.setSize (300,400); tout.setLocation (500,100);
tout.setLineWrap(true); tout.setEditable(false); c.add(tout);
```

```
res = new JLabel (" "); res.setSize(500,25); res.setLocation(100,300); C.add(rs);
resadd = new JTextField(); resadd.setSize(500,25); resadd.setLocation(550,175);
resadd.setLineWrap(true); C.add(resadd); setVisible(true);
}
public void actionPerformed (ActionEvent e)
{ if (e.getSource() == sub)
  { if (term.isSelected())
    { String data1; data=
String data = "Name:" + tname.getText() + "\n" + "Mobile" + tmno.getText(); + "\n";
  if (male.isSelected())
      data1 = "Gender: Male" + "\n";
  else
      data1 = "Gender: Female" + "\n";
String data2
= "DOB:" + (String)date.getSelectedItem()
+ "/" + (String)month.getSelectedItem() + "/"
+ (String)year.getSelectedItem() + "\n";
String data3 = "Address:" + tdd.getText();
tout.setText(data+data1+data2+data3);
tout.setEditable(false); }
else { tout.setText(" "); resadd.setText(" ");
res.setText("please accept the trims & condi
tions");
} }
else if (e.getSource() == reset) {
String def = " "; tname.setText(def);
tadd.setText(def); tmno.setText(def);
res.setText(def); tout.setText(def);
term.setSelected(false); date.setSelectedIndex(0);
month.setSelectedIndex(0); year.setSelectedIndex(0);
resadd.setText(def); } }
class Registration {
public static void main (String[] args) throws Exception
{ Myframe f = new Myframe(); } }
```

output:



CMR GROUP OF INSTITUTIONS
EXPLORE TO INVENT

(7) Explain any four SWING components with suitable program.

JFrame: That represents the main window of a java swing application.

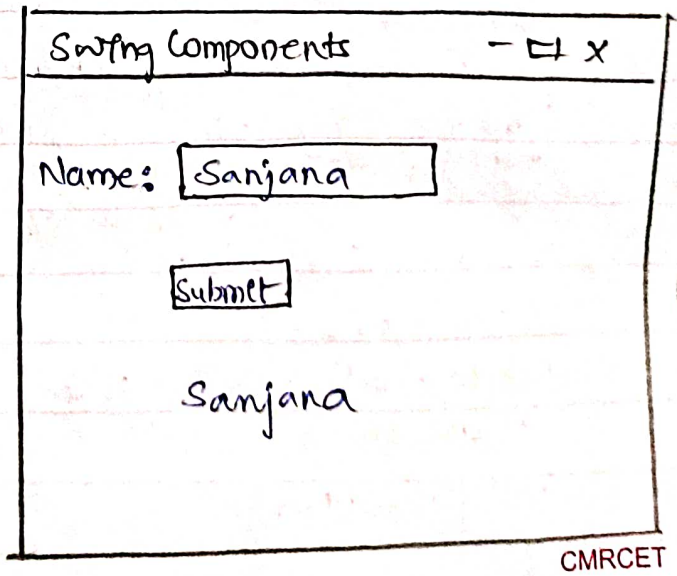
JButton: Represents a button that can trigger action when clicked.

JTextfield: Allows user to enter and edit a single line of text.

JLabel: Display a single line read only text or an Image.

```
import javax.swing.*; import java.awt.*;
import java.awt.event.*;
public class Example extends JFrame
{ Public static void main(String[] args)
{ JFrame f = new JFrame("Swing components");
JLabel l = new JLabel("Name:");
JTextfield t = new JTextfield();
JButton b = new JButton("submit");
b.addActionListener(new ActionListener()
{@Override
  public void actionPerformed(ActionEvent e)
{ String s = t.getText();
JOptionPane.showMessageDialog(f,s);
}});
f.setLayout(new FlowLayout());
f.addJLabel.f.add(l);
f.add(t); f.add(b);
f.setSize(300,150);
f.setDefaultCloseOperation(JFrame.
   EXIT_ON_CLOSE);
f.setVisible(true);
}}
```

output:



CMR GROUP OF INSTITUTIONS

(8) Explain MouseMotionListener, MouseListener with suitable program.

```
Class e
import java.awt.*;
import javax.swing.*; import java.awt.event.*;
class example extends JFrame implements MouseMotionListener, MouseListener
{ Public static void main(String[] args){
JFrame f = new JFrame("MouseMotionListener");
JButton b = new JButton("Mouse");
b.addActionListener(new ActionListener()
{ @override
  public void mouseClicked(MouseEvent e){}
  @override
  public void mousePressed(MouseEvent e)
  { JOptionpane.showMessageDialog(f, "Mouse Pressed"); }
  @override
  public void mouseReleased(MouseEvent e){}
  @override
  public void mouseEntered(MouseEvent e){}
  @override
  public void mouseExited(MouseEvent e){}
});
JLabel l = new JLabel("Move the mouse");
l.addMouseMotionListener(new MouseMotionListener()
{ @override
  public void mouseMoved(MouseEvent e){}
  @override
  public void mouseDragged(Mouse event e)
{ int x = e.getX(); int y = e.getY();
Graphics g = getGraphics(); g.setColor(Color.RED)
f.setText("+x+","+y+")"); }});
f.add(b); f.add(l);
f.setSize(400,200);
f.setDefaultCloseOperation(JFrame.
EXITONCLOSE);
f.setVisible(true);
}}
```

output: