Commands to run in Commnd prompt :-

Save file in desktop new folder java with parent class name.java
Open cd..
cd desktop
cd java
javac filename.java
java filename

Week 9

1. Write a java program to create Thread by extending the Thread class.

Ans :

```
class Multi extends Thread{
    public void run(){
        System.out.println("thread is running...");
        }
        public static void main(String args[]){
        Multi t1=new Multi();
        t1.start();
         }
        }
```

2 Write a java program to create Thread by implementing the Runnable interface

```
Public class ExampleClass implements Runnable {

 @Override
```

```
Public void run() {
System.out.println(“Thread has ended”);
}

Public static void main(String[] args) {
ExampleClass ex = new ExampleClass();
Thread t1= new Thread(ex);
T1.start();
System.out.println("Hi");
}
}
```

3. Write a java program to implement the yield( ) method in thread programming.

```
public class JavaYieldExp extends Thread
{
   public void run()
   {
      for (int i=0; i<3 ; i++)
         System.out.println(Thread.currentThread().getName() + " in control");
   }
   public static void main(String[]args)
   {
      JavaYieldExp t1 = new JavaYieldExp();
      JavaYieldExp t2 = new JavaYieldExp();
```

```
        // this will call run() method
        t1.start();
        t2.start();
        for (int i=0; i<3; i++)
        {
            // Control passes to child thread
            t1.yield();
            System.out.println(Thread.currentThread().getName() + " in control");
        }
    }
}
```

4. Write a java program to implement the sleep(n) method in thread programming.

```
// Java Program for sleeping the main thread.

import java.io.*;
import java.lang.Thread;

class GFG {
    public static void main(String[] args)
    {
        // we can also use throws keyword followed by
        // exception name for throwing the exception

        try {
            for (int i = 0; i < 5; i++) {

                // it will sleep the main thread for 1 sec
                // ,each time the for loop runs
                Thread.sleep(1000);

                // printing the value of the variable
                System.out.println(i);
            }
```

```
                }
                catch (Exception e) {

                        // catching the exception
                        System.out.println(e);
                }
        }
}
```

5. write a java program to implement the suspend( ) & resume( ) method in thread programming.

```
class MyThread implements Runnable {
  public void run() {
    try {
      Thread.sleep(500);
      System.out.println(Thread.currentThread().getName());
    } catch (Exception e) {

    }
  }
}

public class Main {
  public static void main(String[] args) {
    Thread t1 = new Thread(new MyThread());
    Thread t2 = new Thread(new MyThread());
    Thread t3 = new Thread(new MyThread());

    t1.start();
    t2.start();
    t2.suspend();
    t3.start();
    t2.resume();
  }
}
```

7. Create 4 threads with priority 1,3,5,7 respectively. Update a counter in each of the threads for 10 ms. Print the final value of count for each thread.

```
// Java Program to Illustrate Priorities in Multithreading
// via help of getPriority() and setPriority() method

// Importing required classes
import java.lang.*;

// Main class
class ThreadDemo extends Thread {

	// Method 1
	// run() method for the thread that is called
	// as soon as start() is invoked for thread in main()
	public void run()
	{
		// Print statement
		System.out.println("Inside run method");
	}

	// Main driver method
	public static void main(String[] args)
	{
		// Creating random threads
		// with the help of above class
		ThreadDemo t1 = new ThreadDemo();
		ThreadDemo t2 = new ThreadDemo();
		ThreadDemo t3 = new ThreadDemo();

		// Thread 1
		// Display the priority of above thread
		// using getPriority() method
		System.out.println("t1 thread priority : "
						+ t1.getPriority());

		// Thread 1
		// Display the priority of above thread
		System.out.println("t2 thread priority : "
						+ t2.getPriority());

		// Thread 3
		System.out.println("t3 thread priority : "
						+ t3.getPriority());
	    System.out.println("t4 thread priority : "
```

```
                                        + t4.getPriority());

                // Setting priorities of above threads by
                // passing integer arguments
                t1.setPriority(1);
                t2.setPriority(3);
                t3.setPriority(5);
               t4.setPriority(7);

                // t3.setPriority(21); will throw
                // IllegalArgumentException

                // 2
                System.out.println("t1 thread priority : "
                                        + t1.getPriority());

                // 5
                System.out.println("t2 thread priority : "
                                        + t2.getPriority());

                // 8
                System.out.println("t3 thread priority : "
                                        + t3.getPriority());

                // Main thread

                // Displays the name of
                // currently executing Thread
                System.out.println(
                        "Currently Executing Thread : "
                        + Thread.currentThread().getName());

                System.out.println(
                        "Main thread priority : "
                        + Thread.currentThread().getPriority());

                // Main thread priority is set to 10
                Thread.currentThread().setPriority(10);

                System.out.println(
                        "Main thread priority : "
                        + Thread.currentThread().getPriority());
        }
}
```

8. Write a Java Program to Synchronize the Threads Acting on the Same Object. The Synchronized Block in the Program can be Executed by Only One Thread at a Time.

```
// A Java program to demonstrate working of
// synchronized.

import java.io.*;
import java.util.*;

// A Class used to send a message
class Sender
{
	public void send(String msg)
	{
		System.out.println("Sending\t" + msg );
		try
		{
			Thread.sleep(1000);
		}
		catch (Exception e)
		{
			System.out.println("Thread interrupted.");
		}
		System.out.println("\n" + msg + "Sent");
	}
}

// Class for send a message using Threads
class ThreadedSend extends Thread
{
	private String msg;
	Sender sender;

	// Receives a message object and a string
	// message to be sent
	ThreadedSend(String m, Sender obj)
	{
		msg = m;
		sender = obj;
```

```
        }

        public void run()
        {
                // Only one thread can send a message
                // at a time.
                synchronized(sender)
                {
                        // synchronizing the send object
                        sender.send(msg);
                }
        }
}

// Driver class
class SyncDemo
{
        public static void main(String args[])
        {
                Sender send = new Sender();
                ThreadedSend S1 =
                        new ThreadedSend( " Hi " , send );
                ThreadedSend S2 =
                        new ThreadedSend( " Bye " , send );

                // Start two threads of ThreadedSend type
                S1.start();
                S2.start();

                // wait for threads to end
                try
                {
                        S1.join();
                        S2.join();
                }
                catch(Exception e)
                {
                        System.out.println("Interrupted");
                }
        }
}
```

9. Write a Java Program to Check a Thread is Alive or Not

```
// Java program to Illustrate isAlive() Method
// of Thread class

// Main class extending Thread class
public class oneThread extends Thread {

	// Method 1
	// run() method for thread
	public void run()
	{

		// Print statement
		System.out.println("geeks ");

		// Try block to check for exceptions
		try {

			// making thread to sleep for 300 nano-seconds
		// using sleep() method
			Thread.sleep(300);
		}

		// Catch block to handle InterruptedException
		catch (InterruptedException ie) {
		}

		// Display message when exception occurred
		System.out.println("forgeeks ");
	}

	// Method 2
	// Main driver method
	public static void main(String[] args)
	{
		// Creating threads using above class as
		// it is extending Thread class
		oneThread c1 = new oneThread();
		oneThread c2 = new oneThread();
```

```
                // Starting threads
                c1.start();
                c2.start();

                // Checking whether thread is alive or not
                // Returning boolean true if alive else false
                System.out.println(c1.isAlive());
                System.out.println(c2.isAlive());
        }
}
```

10. Write a Java Program to Get the Name of a Running Thread.

```
public class TwoThreadGetName extends Thread {
  public void run() {
    for (int i = 0; i < 10; i++) {
      printMsg();
    }
  }
  public void printMsg() {
    Thread t = Thread.currentThread();
    String name = t.getName();
    System.out.println("name=" + name);
  }
  public static void main(String[] args) {
    TwoThreadGetName tt = new TwoThreadGetName();
    tt.start();
    for (int i = 0; i < 10; i++) {
      tt.printMsg();
    }
  }
}
```

Week 10

1) **Implement concept of Applet**

```
Import java.applet.Applet;
Import java.awt.Graphics;
Public class First extends Applet{

Public void paint(Graphics g){
g.drawString(“welcome to applet”,150,150);
}
}
```

My applet.html

```
<html>
<body>
<applet code=”First.class” width=”300” height=”300”>
</applet>
</body>
</html>
```

2) **Parameter passing in Applet**

```
Import java.applet.Applet;
Import java.awt.Graphics;
Public class UseParam extends Applet{
Public void paint(Graphics g){
String str=getParameter(“msg”);
g.drawString(str,50, 50);
}
 }
```

3) J button class

```
Import javax.swing.*;
Public class ButtonExample {
Public static void main(String[] args) {
   JFrame f=new JFrame("Button Example");
   JButton b=new JButton("Click Here");
   b.setBounds(50,100,95,30);
   f.add(b);
   f.setSize(400,400);
   f.setLayout(null);
   f.setVisible(true);
}
}
```

4) J Texfield class

```
Import javax.swing.*;
Class TextFieldExample
{
Public static void main(String args[])
   {
   JFrame f= new JFrame("TextField Example");
   JTextField t1,t2;
   T1=new JTextField("Welcome to Javatpoint.");
   T1.setBounds(50,100, 200,30);
   T2=new JTextField("AWT Tutorial");
   T2.setBounds(50,150, 200,30);
   f.add(t1); f.add(t2);
   f.setSize(400,400);
   f.setLayout(null);
   f.setVisible(true);
   }
   }
```

5) J Pannel class

```
Import java.awt.*;
Import javax.swing.*;
Public class PanelExample {
    PanelExample()
      {
```

```
JFrame f= new JFrame("Panel Example");
JPanel panel=new JPanel();
Panel.setBounds(40,80,200,200);
Panel.setBackground(Color.gray);
JButton b1=new JButton("Button 1");
B1.setBounds(50,100,80,30);
B1.setBackground(Color.yellow);
JButton b2=new JButton("Button 2");
B2.setBounds(100,100,80,30);
B2.setBackground(Color.green);
Panel.add(b1); panel.add(b2);
f.add(panel);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
Public static void main(String args[])
{
New PanelExample();
}
}
```

6) J Menu class

```
Import javax.swing.*;
Class MenuExample
{
    JMenu menu, submenu;
    JMenuItem i1, i2, i3, i4, i5;
    MenuExample(){
    JFrame f= new JFrame("Menu and MenuItem Example");
    JMenuBar mb=new JMenuBar();
    Menu=new JMenu("Menu");
    Submenu=new JMenu("Sub Menu");
    I1=new JMenuItem("Item 1");
    I2=new JMenuItem("Item 2");
    I3=new JMenuItem("Item 3");
    I4=new JMenuItem("Item 4");
    I5=new JMenuItem("Item 5");
    Menu.add(i1); menu.add(i2); menu.add(i3);
    Submenu.add(i4); submenu.add(i5);
    Menu.add(submenu);
    Mb.add(menu);
```

```
        f.setJMenuBar(mb);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
}
Public static void main(String args[])
{
New MenuExample();
```

}}

7. ChatFrame comprising JFrame, JMenuBar, JMenu, JMenuItem, JPanel, JLabel, JTextField, JButton etc.

CODE:

```
import java.awt.*;
import javax.swing.*;
public class ChatFrame extends JFrame {
    private JMenuBar menuBar;
    private JMenu fileMenu, editMenu, helpMenu;
    private JMenuItem newMenuItem, openMenuItem, saveMenuItem,
                      exitMenuItem;
    private JMenuItem cutMenuItem, copyMenuItem, pasteMenuItem;
    private JMenuItem aboutMenuItem;
    private JPanel chatPanel, ~~chatLabel~~
    private JLabel chatLabel;
    private JTextField chatTextField;
    private JButton chatButton;
    public ChatFrame() {
        setTitle("Chat Frame");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400, 300); setLayout(new BorderLayout());
        menuBar = new JMenuBar();
        fileMenu = new JMenu("File");
        newMenuItem = new JMenuItem("New");
        openMenuItem = new JMenuItem("Open");
        saveMenuItem = new JMenuItem("Save");
        exitMenuItem = new JMenuItem("Exit");
        fileMenu.add(newMenuItem); fileMenu.add(openMenuItem);
        fileMenu.addSeparator(); fileMenu.add(exitMenuItem);
        editMenu = new JMenu("Edit");
        cutMenuItem = new JMenuItem("Cut");
        copyMenuItem = new JMenuItem("Copy");
        pasteMenuItem = new JMenuItem("Paste");
        editMenu.add(cutMenuItem); ~~editMenu~~
        editMenu.add(copyMenuItem); editMenu.add(pasteMenuItem);
        helpMenu = new JMenu("Help");
        aboutMenuItem = new JMenuItem("About"); helpMenu.add(about
                                                MenuItem);
        menubar.add(fileMenu);
        menubar.add(editMenu); menubar.add(helpMenu);
        chatPanel = new JPanel(); chatPanel.setLayout(new FlowLayout());
        chatLabel = new JLabel("Chat:"); chatTextField = new JTextField(20);
        chatButton = new JButton("Send");
        chatPanel.add(chatLabel); chatPanel.add(chatTextField);
        chatPanel.add(chatButton);
        set JMenuBar(menuBar);
        add(chatPanel, BorderLayout.CENTER);
        setVisible(true);
    }
    public static void main(String args[]) {
        new ChatFrame();
    }
}
```