JAVA PROJECT:

GUI :

```
import java.lang.reflect.Array;
import java.util.List;

import org.eclipse.swt.SWT;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Text;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.events.MouseAdapter;
import org.eclipse.swt.events.MouseEvent;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

public class bankinput {

	protected Shell shell;
	private Text text;
	 static bank[] a=new bank[1000];
	int i=0;
	private Text text_1;
	private Button btnNewButton_1;
	private Button btnAgian;

	/**
	 * Launch the application.
	 * @param args
	 */
	public static void main(String[] args) {

		try {
			bankinput window = new bankinput();
			window.open();
		} catch (Exception e) {
			e.printStackTrace();
		}

		int sum =0;
		for(int i=0;i<2*bank.getuser();i++) {
			if(a[i].getdeparttime()<a[i].getintime()) {
				System.out.println("Processing an Deaprture event at time
:"+a[i].getdeparttime());
				System.out.println("Processing an Arrival event at time
:"+a[i].getintime());
			}
			else if(a[i].getdeparttime()>=a[i].getintime()) {
				System.out.println("Processing an Arrival event at time
:"+a[i].getintime());
				System.out.println("Processing an Deaprture event at time
:"+a[i].getdeparttime());
			}
```

```
                sum=sum+a[i].gettranstime();
            }
            System.out.println("The total no of people processed :
"+bank.getuser());
            System.out.println("Average amount of time wasting :
"+(sum/bank.getuser()));
      }


      /**
       * Open the window.
       * @wbp.parser.entryPoint
       */
      public void open() {
            Display display = Display.getDefault();
            createContents();
            shell.open();
            shell.layout();
            while (!shell.isDisposed()) {
                  if (!display.readAndDispatch()) {
                        display.sleep();
                  }
            }
      }

      /**
       * Create contents of the window.
       */
      protected void createContents() {
            shell = new Shell();
            shell.setSize(450, 300);
            shell.setText("SWT Application");

            text = new Text(shell, SWT.BORDER);
            text.setBounds(37, 107, 92, 21);

            Button btnNewButton = new Button(shell, SWT.NONE);
            btnNewButton.setBounds(10, 46, 131, 25);
            btnNewButton.setText("Enter Arrival time:");

            Button btnEnter = new Button(shell, SWT.NONE);
            btnEnter.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseDoubleClick(MouseEvent e) {
                        a[i]=new bank();
                        String val=text.getText();
                        String val1=text_1.getText();
                              a[i].setintime(Integer.parseInt(val));
                              bank.setuser();
                              a[i].settranstime(Integer.parseInt(val1));
                              a[i].calculate();
                              i++;

                  }
```

```
            });
            btnEnter.setBounds(132, 169, 75, 25);
            btnEnter.setText("Enter.");

            Button btnEnd = new Button(shell, SWT.NONE);
            btnEnd.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseDoubleClick(MouseEvent e) {
                        shell.close();
                  }
            });
            btnEnd.setBounds(271, 236, 75, 25);
            btnEnd.setText("END.");

            text_1 = new Text(shell, SWT.BORDER);
            text_1.setBounds(217, 107, 103, 21);

            btnNewButton_1 = new Button(shell, SWT.NONE);
            btnNewButton_1.setBounds(197, 46, 134, 25);
            btnNewButton_1.setText("Enter Transaction Time:");

            btnAgian = new Button(shell, SWT.NONE);
            btnAgian.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseDoubleClick(MouseEvent e) {

                        text_1.setText("");
                        text.setText("");

                  }
            });

            btnAgian.setBounds(10, 236, 75, 25);
            btnAgian.setText("Again?");

      }
}
```

Stimulation Class:

```
public class SIMULATION {
    private PLANT plant=new PLANT();
    private WEATHER weather=new WEATHER();
    private SoilWater soilWater=new SoilWater();
    public void Initialization(int count){
        if(count==0){
            weather.Initialization(new File("WEATHER_IN.txt"));
        }
        if(count==1){
            plant.Initialization(new File("PLANT_IN.txt"));
        }
        if(count==2){
```

```
            soilWater.Initialization(new File("SOIL_IN.txt"));
        }
    }
    public void rateCalculation(int count){
        if(count==0){
            weather.rateCalculation();
        }
        if(count==1) {
            plant.rateCalculation();
        }
        if(count==2) {
            soilWater.rateCalculation();
        }
    }
    public void Integration(int count){
        if(count==1) {
            plant.Integration();
        }
        if (count==2) {
            soilWater.Integration();
        }
    }
    public void output(){
        plant.output();
        soilWater.output();
    }
    public void close(){
        plant.close();
        weather.close();
        soilWater.close();
    }

}
```

Plant Class:

```
import java.io.*;
import java.util.Scanner;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class PLANT {
    private WEATHER weather= new  WEATHER();
    private File PLANT_IN;
    private double PT,PG,PD,ROW_SPC,DIAT,
            dn,rm,di,tb,swfac,EMP1,EMP2,nb,N,LAI,P1,SLA,dwc,dwr,
            dwf,dw,fc,intot,w,wc,wr,wf,Lfmax,dLAI;
    private static double Int=0;
    private int veg_days=0;
    private int rep_days=0;
    private static Boolean matured=false;

    public void setDi(double di) {
        this.di = di;
```

```
}

public void setDIAT(double DIAT) {
    this.DIAT = DIAT;
}

public void setDn(double dn) {
    this.dn = dn;
}

public void setDw(double dw) {
    this.dw = dw;
}

public void setEMP1(double EMP1) {
    this.EMP1 = EMP1;
}

public void setEMP2(double EMP2) {
    this.EMP2 = EMP2;
}

public void setPD(double PD) {
    this.PD = PD;
}

public void setLAI(double LAI) {
    this.LAI = LAI;
}

public void setN(double n) {
    N = n;
}

public void setPG(double PG) {
    this.PG = PG;
}

public void setDwc(double dwc) {
    this.dwc = dwc;
}

public void setNb(double nb) {
    this.nb = nb;
}

public void setDwf(double dwf) {
    this.dwf = dwf;
}

public void setDwr(double dwr) {
    this.dwr = dwr;
}

public void setFc(double fc) {
```

```
    this.fc = fc;
}

public void setP1(double P1) {
    this.P1 = P1;
}

public void setPT(double PT) {
    this.PT = PT;
}

public void setRm(double rm) {
    this.rm = rm;
}

public void setROW_SPC(double ROW_SPC) {
    this.ROW_SPC = ROW_SPC;
}

public void setSLA(double SLA) {
    this.SLA = SLA;
}

public void setIntot(double intot) {
    this.intot = intot;
}

public void setSwfac(double swfac) {
    this.swfac = swfac;
}

public void setTb(double tb) {
    this.tb = tb;
}

public void setW(double w) {
    this.w = w;
}

public void setWc(double wc) {
    this.wc = wc;
}

public void setWr(double wr) {
    this.wr = wr;
}

public void setLfmax(double lfmax) {
    this.Lfmax = lfmax;
}

public double getDi() {
    return di;
}
```

```
public double getDIAT() {
    return DIAT;
}

public double getDn() {
    return dn;
}

public double getDw() {
    return dw;
}

public double getDwc() {
    return dwc;
}

public double getPD() {
    return PD;
}

public double getDwf() {
    return dwf;
}

public double getDwr() {
    return dwr;
}

public double getEMP1() {
    return EMP1;
}

public double getEMP2() {
    return EMP2;
}

public double getFc() {
    return fc;
}
   public double getWf() {
          return wf;
   }

   public void setWf(double wf) {
          this.wf = wf;
   }

public double getIntot() {
    return intot;
}

public double getLAI() {
    return LAI;
}
```

```
public double getN() {
    return N;
}

public double getNb() {
    return nb;
}

public double getPG() {
    return PG;
}

public double getP1() {
    return P1;
}

public double getPT() {
    return PT;
}

public double getRm() {
    return rm;
}

public double getROW_SPC() {
    return ROW_SPC;
}

public double getSLA() {
    return SLA;
}

public double getSwfac() {
    return swfac;
}

public double getTb() {
    return tb;
}

public double getW() {
    return w;
}

public double getWc() {
    return wc;
}

public double getWr() {
    return wr;
}

public double getLfmax() {
    return Lfmax;
}
```

```
    private void PTS(){
        PT=1-0.0025*Math.pow(0.25* weather.getTmin()+0.75* weather.getTmax()-26,2);
    }
    private void LAIS() {
        double a;
        if(N<=Lfmax) {
            dn = rm * PT;
            N +=dn;
            a = Math.exp(EMP2 * (N - nb));
            System.out.println(swfac);
            dLAI = swfac * PT * PD * EMP1 * dn * (a / (1 + a));
            veg_days++;
        }
        else {
            di = ((weather.getTmax() + weather.getTmin()) / 2) - tb;
            Int += di;
            System.out.println(Int);
            dLAI = dLAI - (PD * di * SLA * P1);
            rep_days++;
        }
        double w2;
        w2=PG/2;
        dwc=w2-wc;
        dwr=w2-wr;
        dwf=PG;
        dw=dwf+dwr+dwc;
        dw*=PD;
    }


    private void PGS(){
    double y1=1.5-0.768*Math.pow((Math.pow(( ROW_SPC*0.01),2)*PD),0.1);
    PG=(weather.getSRAD()/PD)*(1.0-Math.exp(-y1*LAI));
    }
    public void Initialization(File PLANT_IN){
        System.out.println("Plant's Initialization called");
        try{
            this.PLANT_IN=PLANT_IN;
            Scanner scanner = new Scanner(this.PLANT_IN);
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                // Use regular expressions to match the variable name and value
                Matcher matcher = Pattern.compile("([a-zA-
Z_]+):\\s*(\\S+)").matcher(line);
                if (matcher.find()) {
                    if(matcher.group(1).equals("EMP_ONE")){
                        EMP1=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("Int")){
                        Int=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("EMP_TWO")){
                        EMP2=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("fc")){
```

```
                fc=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("intot")){
                intot=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("LAI")){
                LAI=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("Lfmax")){
                Lfmax=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("N")){
                N=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("nb")){
                nb=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("P1")){
                P1=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("PD")){
                PD=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("rm")){
                rm=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("SLA")){
                SLA=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("tb")){
                tb=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("wc")){
                wc=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("veg_days")){
                veg_days=Integer.parseInt(matcher.group(2));
            }
            if(matcher.group(1).equals("rep_days")){
                rep_days=Integer.parseInt(matcher.group(2));
            }
            if(matcher.group(1).equals("w")){
                w=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("swfac")){
                swfac=Double.parseDouble(matcher.group(2));
            }
            if(matcher.group(1).equals("wr")) {
                wr = Double.parseDouble(matcher.group(2));
            }
        }
    } scanner.close();
}catch (Exception e){
    e.getStackTrace();
}
```

```
    }
    public void rateCalculation(){
        PTS();
        PGS();
        LAIS();
    }
    public void Integration(){
        w+=dw;
        wc+=dwc;
        setWf(getWf() + dwf);
        wr+=dwr;
        LAI+=dLAI;
        N+=dn;
        if(Int>=intot){
            System.out.println("Plant matured!!!");
            matured=true;
        }
    }
    public void output(){
        try{
           FileWriter fileWriter= new FileWriter(this.PLANT_IN);
           PrintWriter printWriter= new PrintWriter(fileWriter);
            printWriter.println("EMP_ONE:" + EMP1);
            printWriter.println("EMP_TWO:" + EMP2);
            printWriter.println("intot:" + intot);
            printWriter.println("fc:" + fc);
            printWriter.println("LAI:" + LAI);
            printWriter.println("w:" + w);
            printWriter.println("wc:" + wc);
            printWriter.println("wr:" + wr);
            printWriter.println("PD:" + PD);
            printWriter.println("Lfmax:" + Lfmax);
            printWriter.println("N:" + N);
            printWriter.println("nb:" + nb);
            printWriter.println("P1:" + P1);
            printWriter.println("rm:" + rm);
            printWriter.println("SLA:" + SLA);
            printWriter.println("tb:" + tb);
            printWriter.println("veg_days:" + veg_days);
            printWriter.println("rep_days:" + rep_days);
            printWriter.println("Int:" + Int);
            printWriter.close();
        }catch (Exception e){
            e.getStackTrace();
        }

    }
    public void close(){
        PLANT_IN=null;

    }
}
```

Weather Class:

```
import java.io.File;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class WEATHER {
    protected static double PAR,RAIN,SRAD,Tmax,Tmin;
    protected static int DATE;
    private File WEATHER_IN;

    public void Initialization(File WEATHER_IN){
        System.out.println("Weather's Initialization called!");
        try{
            Scanner scanner = new Scanner(WEATHER_IN);
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                // Use regular expressions to match the variable name and value
                Matcher matcher = Pattern.compile("([a-zA-
Z_]+):\\s*(\\S+)").matcher(line);
                if (matcher.find()) {
                    if(matcher.group(1).equals("Tmin")){
                        Tmin=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("Tmax")){
                        Tmax=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("RAIN")){
                        RAIN=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("SRAD")){
                        SRAD=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("PAR")){
                        PAR=Double.parseDouble(matcher.group(2));
                    }
                    if(matcher.group(1).equals("DATE")){
                        DATE=Integer.parseInt(matcher.group(2));
                    }
                }
            }
            scanner.close();
        }catch (Exception e){
            e.getStackTrace();
        }
    }

    public void setTmax(double tmax) {
        Tmax = tmax;
    }

    public void setTmin(double tmin) {
        Tmin = tmin;
```

```
    }

    public void setSRAD(double SRAD) {
        this.SRAD = SRAD;
    }

    public void setDATE(int DATE) {
        this.DATE = DATE;
    }

    public void setPAR(double PAR) {
        this.PAR = PAR;
    }

    public void setRAIN(double RAIN) {
        this.RAIN = RAIN;
    }

    public double getTmin() {
        return Tmin;
    }

    public double getTmax() {
        return Tmax;
    }

    public double getSRAD() {
        return SRAD;
    }

    public double getPAR() {
        return PAR;
    }

    public double getRAIN() {
        return RAIN;
    }

    public int getDATE() {
        return DATE;
    }

    public void rateCalculation(){

    }
    public void close(){

    }
}
```

SoilWater Class:

```
import java.io.*;
```

```
import java.io.FileWriter;
import java.io.PrintWriter;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class SoilWater  {
    private WEATHER weather=new WEATHER();
    private PLANT plant= new PLANT();
    private File Soil_IN;

    private double SWFAC1,SWFAC2,ALB,EPA,INF,ST,WP,FC,S,SWC_ADJ,ETP,g,Rn,FCP,
            THE,ROF,DP,WTABLE,DWT,Train,TIRR,TESA,

TEPA,TDRN,TINF,POT_INF,DRN,TROF,IRR,DRNP,ESP,EPP,ESA,CN,WPP,SWC,STP,EEQ,Tmed,a;
    private static int count=0;
    public void RUNOFF(){
        S=254*((100/CN)-1);

        if(count>0){
            if(POT_INF>0.2*S){
                ROF=(Math.pow(POT_INF-0.2*S,2))/(POT_INF+0.8*S);
            }
            else {
                ROF=0;
            }
        }
        count++;
    }
    public void STRESS(){
        THE=WP+0.75*(FC-WP);
        if (SWC < WP) {
                SWFAC1 = 0.0;}
        else if (SWC > THE)
                SWFAC1 = 1.0;
        else{
                SWFAC1 = (SWC - WP) / (THE - WP);
        SWFAC1 = Math.max(Math.min(SWFAC1, 1.0), 0.0);}

    }
    public void RC(){

    }
    public void DRAIN(){
        DRN=(SWC-FC)*DRNP;
    }
    public void ETPS(){

        ALB=0.1*Math.exp(-0.7*plant.getLAI())+0.2*(1-Math.exp(-0.7*plant.getLAI()));
        Tmed=0.6* weather.getTmax()+0.4* weather.getTmin();
        EEQ=weather.getSRAD()*(4.88*Math.pow(10,-3)-4.37*Math.pow(10,-
3)*ALB)*(Tmed+29);
        ETP=a * (Rn / (g + (1 - a) * (Rn)));
        ESP=ETP*Math.exp(-0.7*plant.getLAI());
```

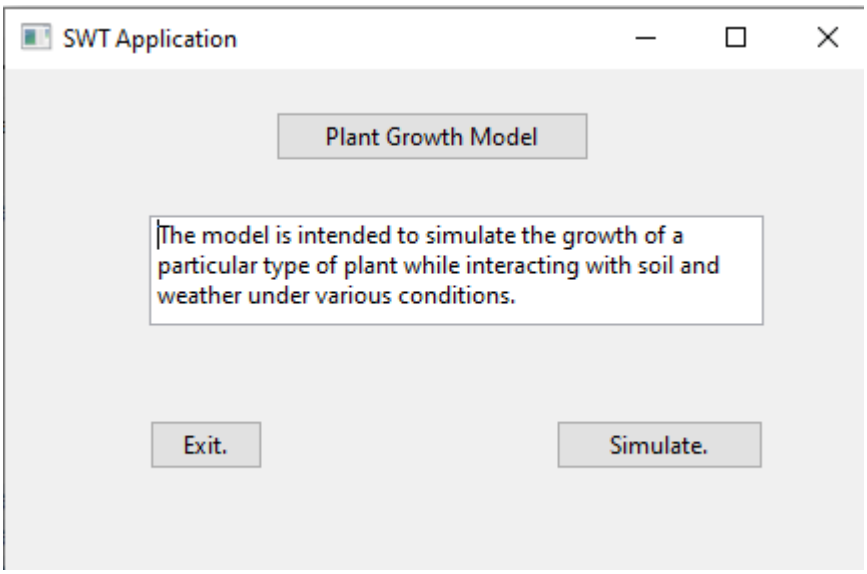
```
        EPP=ETP*(1-Math.exp(-0.7*plant.getLAI()));
    }
    public void ESAS(){
        if(SWC<WP){
            a=0;
        } else if (SWC>FC) {
            a=1;
        }
        else{
            a=(SWC-WP)*(SWC-FC);
        }
        ESA=ESP*a;
        EPA=EPP-SWFAC2;
    }
    public void WBA(){

    }

    public void Initialization(File Soil_IN){
        System.out.println("Soil Water's Initialization called!");
        this.Soil_IN=Soil_IN;

        try {
            Scanner scanner = new Scanner(this.Soil_IN);
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                // Use regular expressions to match the variable name and value
                Matcher matcher = Pattern.compile("([a-zA-
Z_]+):\\s*(\\S+)").matcher(line);
                if (matcher.find()) {
                    if (matcher.group(1).equals("DP")) {
                        DP = Double.parseDouble(matcher.group(2));
//                        System.out.println(DP);
                    }
                    if (matcher.group(1).equals("SWC")) {
                        SWC = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("WPP")) {
                        WPP = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("STP")) {
                        STP = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("CN")) {
                        CN = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("DRNP")) {
                        DRNP = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("FCP")) {
                        FCP = Double.parseDouble(matcher.group(2));
                    }
                    if (matcher.group(1).equals("Rn")) {
                        Rn = Double.parseDouble(matcher.group(2));
                    }
```

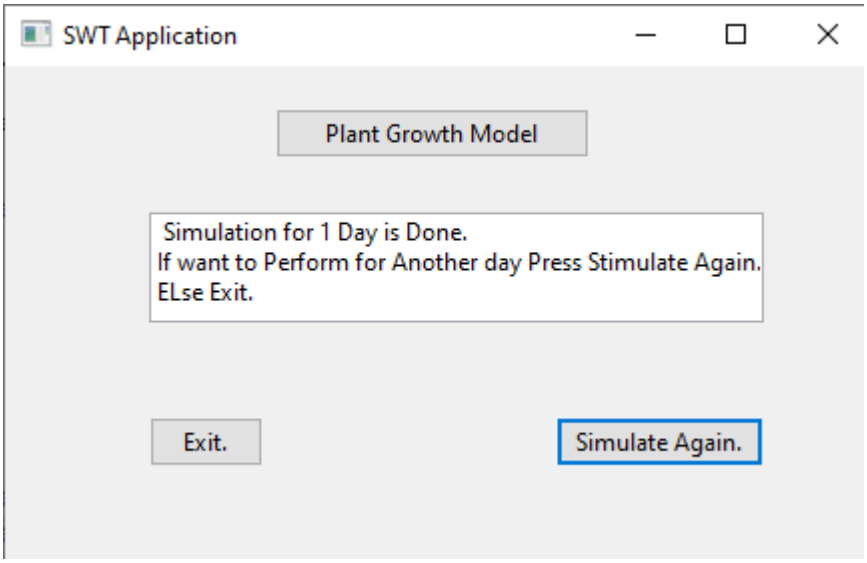
```
                if (matcher.group(1).equals("g")) {
                    g = Double.parseDouble(matcher.group(2));
                    }
                 if (matcher.group(1).equalsIgnoreCase("IRR")) {
                        IRR = Double.parseDouble(matcher.group(2));
                }
            }
        }
        scanner.close();
    }catch (Exception e){
        e.getStackTrace();
    }
    try {
        File file = new File("PLANT_IN.txt");
        FileWriter fileWriter = new FileWriter(file,true);
        PrintWriter printWriter = new PrintWriter(fileWriter);
        printWriter.println("swfac:" + (SWFAC2+SWFAC1/2));
        printWriter.close();
    }catch(Exception e){
        e.getStackTrace();
    }
    WP=DP*WPP*10;
    FC=DP*FCP*10;
    ST=DP*STP*10;
    RUNOFF();
    STRESS();
    Train=0;
    TIRR=0;
    TESA=0;
    TEPA=0;
    TROF=0;
    TDRN=0;
    TINF=0;
}
public void rateCalculation(){


    POT_INF= weather.getRAIN()+IRR;
    TIRR+=IRR;
    Train+= weather.getRAIN();
    DRAIN();
    ROF=0;
    if(POT_INF>0){
        RUNOFF();
    }
    INF=POT_INF-ROF;
    ETPS();
    ESAS();
}
public void Integration(){
    SWC=SWC+(INF-ESA-EPA-DRN);
    if(SWC>ST){
        ROF=ROF+(SWC-ST);
        SWC=ST;
    }
```

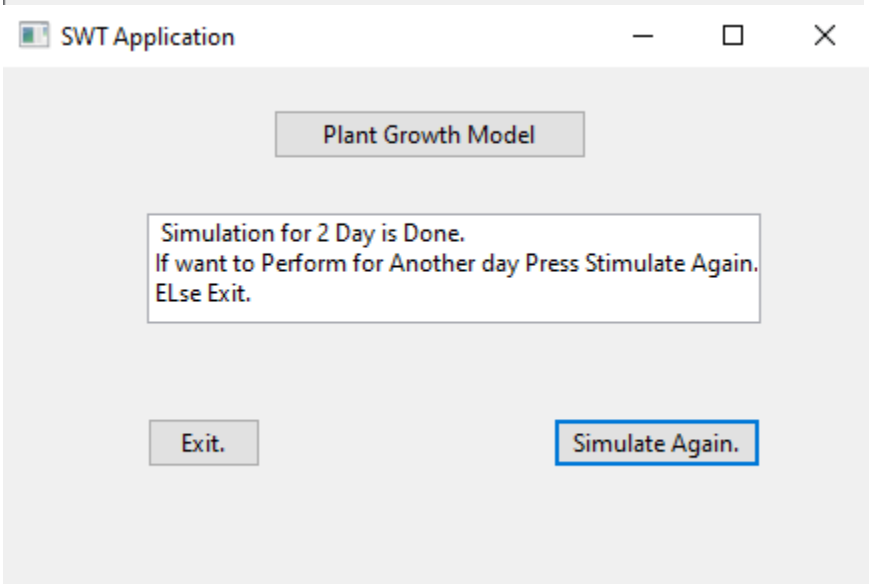
```
    else if(SWC<0){
        SWC_ADJ=SWC_ADJ-SWC;
        SWC=0;
    }
   STRESS();
    WTABLE = (SWC - FC) / (ST - FC) * DP * 10;
    DWT = DP * 10 - WTABLE;
    if (DWT > 250) {
            SWFAC2 = 1.0;}
    else{
            SWFAC2 = DWT / 250;}
}
public void output(){
    try{
        FileWriter fileWriter= new FileWriter(this.Soil_IN);
        PrintWriter printWriter= new PrintWriter(fileWriter);
        printWriter.println("SWFAC_ONE:" + SWFAC1);
        printWriter.println("SWFAC_TWO:" + SWFAC2);
        printWriter.println("ALB:" + ALB);
        printWriter.println("INF:" + INF);
        printWriter.println("ST:" + ST);
        printWriter.println("WP:" + WP);
        printWriter.println("FC:" + FC);
        printWriter.println("S:" + S);
        printWriter.println("SWC_ADJ:" + SWC_ADJ);
        printWriter.println("THE:" + THE);
        printWriter.println("ROF:" + ROF);
        printWriter.println("DP:" + DP);
        printWriter.println("DWT:" + DWT);
        printWriter.println("WTABLE:" + WTABLE);
        printWriter.println("Train:" + Train);
        printWriter.println("TIRR:" + TIRR);
        printWriter.println("TESA:" + TESA);
        printWriter.println("TEPA:" + TEPA);
        printWriter.println("TDRN:" + TDRN);
        printWriter.println("TINF:" + TINF);
        printWriter.println("POT_INF:" + POT_INF);
        printWriter.println("g:" + g);
        printWriter.println("Rn:" + Rn);
        printWriter.println("DRN:" + DRN);
        printWriter.println("TROF:" + TROF);
        printWriter.println("IRR:" + IRR);
        printWriter.println("DRNP:" + DRNP);
        printWriter.println("ESP:" + ESP);
        printWriter.println("EPP:" + EPP);
        printWriter.println("ESA:" + ESA);
        printWriter.println("CN:" + CN);
        printWriter.println("WPP:" + WPP);
        printWriter.println("SWC:" + SWC);
        printWriter.println("FCP:" + FCP);
        printWriter.println("EEQ:" + EEQ);
        printWriter.println("TMED:" + Tmed);
        printWriter.println("a:" + a);
        printWriter.close();
    }catch (Exception e){
```

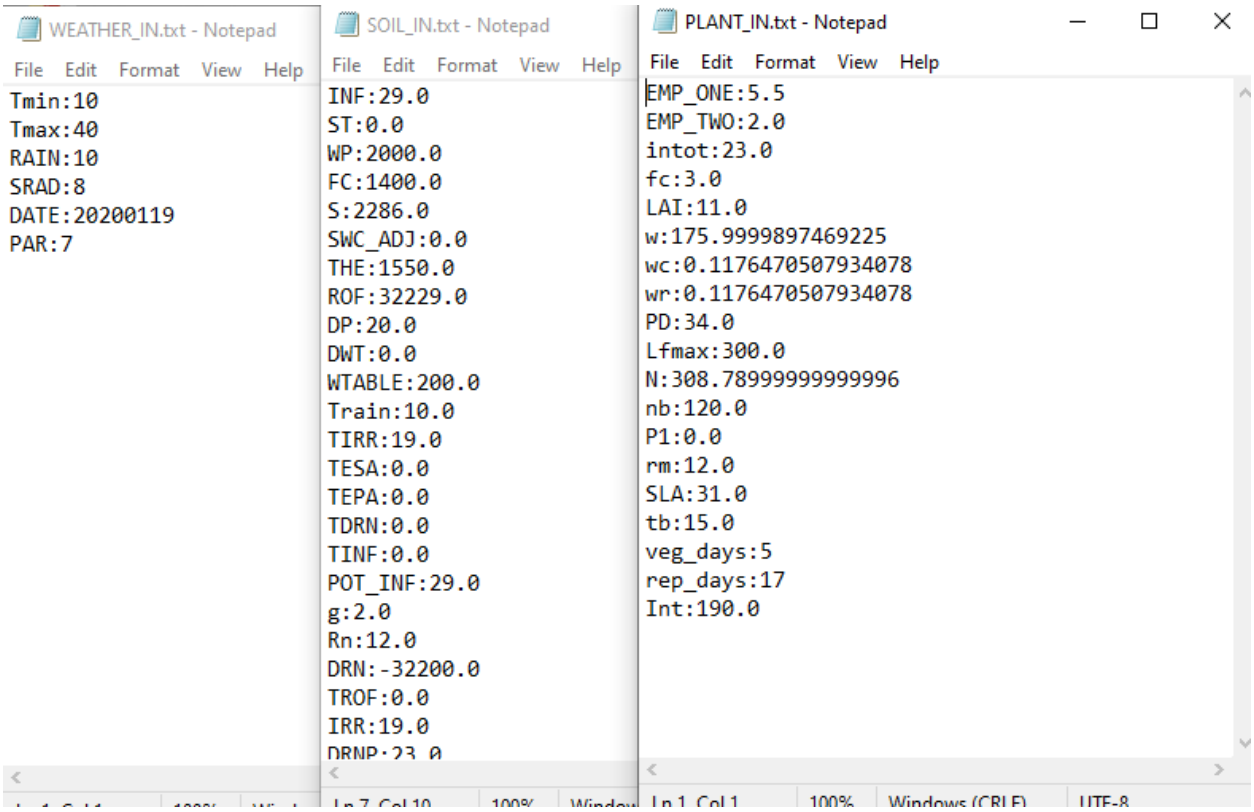
```
            e.getStackTrace();
        }

    }
    public void close(){
        Soil_IN=null;

    }

}
```

Output:




SWT Application
Plant Growth Model
Simulation for 1 Day is Done.
If want to Perform for Another day Press Stimulate Again.
ELse Exit.
Exit.
Simulate Again.


SWT Application
Plant Growth Model
Simulation for 2 Day is Done.
If want to Perform for Another day Press Stimulate Again.
ELse Exit.
Exit.
Simulate Again.



WEATHER_IN.txt - Notepad

File Edit Format View Help

```
Tmin:10
Tmax:40
RAIN:10
SRAD:8
DATE:20200119
PAR:7
```

SOIL_IN.txt - Notepad

File Edit Format View Help

```
INF:29.0
ST:0.0
WP:2000.0
FC:1400.0
S:2286.0
SWC_ADJ:0.0
THE:1550.0
ROF:32229.0
DP:20.0
DWT:0.0
WTABLE:200.0
Train:10.0
TIRR:19.0
TESA:0.0
TEPA:0.0
TDRN:0.0
TINF:0.0
POT_INF:29.0
g:2.0
Rn:12.0
DRN:-32200.0
TROF:0.0
IRR:19.0
```

PLANT_IN.txt - Notepad

File Edit Format View Help

```
EMP_ONE:5.5
EMP_TWO:2.0
intot:23.0
fc:3.0
LAI:11.0
w:175.9999897469225
wc:0.1176470507934078
wr:0.1176470507934078
PD:34.0
Lfmax:300.0
N:308.78999999999996
nb:120.0
P1:0.0
rm:12.0
SLA:31.0
tb:15.0
veg_days:5
rep_days:17
Int:190.0
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8