CS F213

Object Oriented

Programming

Assignment

|  |  |
| --- | --- |
| Name | ID no. |
|  |  |
| Vansh Jain | 2020A7PS0079U |
|  |  |

**Birla Institute of Technology and Science**

**D.I.A.C, Dubai, UAE**

**A Report**

**on**

**Temperature converter on java**

**Prepared for**

**Dr. Pranav Mothabhau Pawar**

**Instructor in charge**

**By**

|  |  |
| --- | --- |
| Name | ID no. |
| Vansh Jain | 2020A7PS0079U |

**Approved by**

**Dr. Pranav Mothabhau Pawar**

**Instructor in charge**

**December 2021**

**Abstract**

The main goal of this project is to understand the concepts Object-Oriented Programming and use Java to implement it with the help of a mathematical background. The project “Temperature Conversion System” is developed in java, which mainly focuses on conversion to and from Celsius, Fahrenheit, and Kelvin. Java in-built packages like swing and awt are utilized. We have also discussed the various applications of OOP in real life

**Acknowledgement**

I would like to express my gratitude towards, Dr. Pranav Mothabhau Pawar the instructor of the course on discrete structures, for this opportunity to increase my understanding on the concepts of Object-Oriented Programming and usage of the Java language to code for programs related to graphs. I would also like to thank Dr. Sujala D. Shetty the instructor-in-charge for her guidance and support throughout the project.

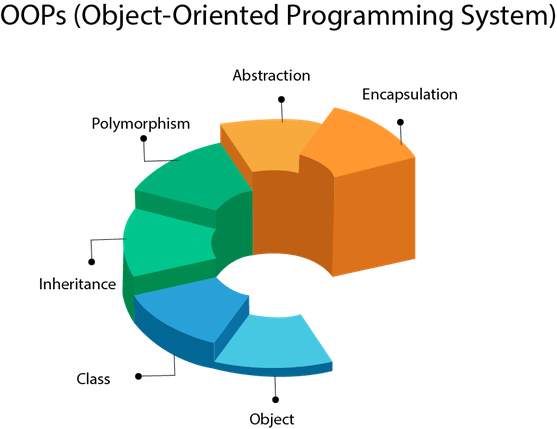
**Table of contents**

|  |
| --- |
| **Abstract** |
| **Acknowledgement** |
| **List of figures** |
| 1. **Introduction**    1. **Aim**    2. **Object-oriented programming (OOP)**       1. **Objects**       2. **Classes**       3. **Methods**       4. **Encapsulation**       5. **Abstraction**       6. **Inheritance**       7. **Polymorphism** 2. **Software used**    1. **Packages used**       1. **Swing**       2. **AWT** 3. **Source code**    1. **Displayed output**    2. **Concepts used**        1. **GUI**       2. **Class**       3. **Objects**       4. **Inheritance**       5. **Abstraction**       6. **Polymorphism** 4. **Applications of the program**    1. **Units of Temperature**       1. **Celsius**       2. **Fahrenheit**       3. **Kelvin** 5. **Conclusion**    1. **Software** 6. **References** |

1. **Introduction**
   1. **Aim-**

Create a temperature conversion java application using the various concepts of OOP learnt throughout the course.

* 1. **Object-oriented programming (OOP)**

OOP is a computer programming model that organizes software design around data, or objects, rather than functions and logic. OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for programs that are large, complex, and actively updated or maintained.

* + 1. **Objects**

An object can be defined as a data field that has unique attributes and behavior. Objects are instances of a class created with specifically defined data. Objects can correspond to real-world objects or an abstract entity. When class is defined initially, the description is the only object that is defined.

* + 1. **Classes**

Classes are user-defined data types that act as the blueprint for individual objects, attributes, and methods.

* + 1. **Methods**

Methods are functions that are defined inside a class that describe the behaviors of an object. Each method contained in class definitions starts with a reference to an instance object. Additionally, the subroutines contained in an object are called instance methods. Programmers use methods for reusability or keeping functionality encapsulated inside one object at a time.

* + 1. **Encapsulation**

The practice of keeping fields within a class private, then providing access to those fields via public methods. Encapsulation is a protective barrier that keeps the data and code safe within the class itself. We can then reuse objects like code components or variables without allowing open access to the data system wide.

* + 1. **Abstraction**

Using simple things to represent complexity. We all know how to turn the TV on, but we don’t need to know how it works to enjoy it. In Java, abstraction means simple things like objects, classes and variables represent more complex underlying code and data. This is important because it lets you avoid repeating the same work multiple times.

* + 1. **Inheritance**

A special feature of Object-Oriented Programming in Java, Inheritance lets programmers create new classes that share some of the attributes of existing classes. Using Inheritance lets us build on previous work without reinventing the wheel.

* + 1. **Polymorphism**

Allows programmers to use the same word in Java to mean different things in different contexts. One form of polymorphism is method overloading. That’s when the code itself implies different meanings. The other form is method overriding. That’s when the values of the supplied variables imply different meanings. Let’s delve a little further.

1. **Software used**

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere, meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture.

* 1. **Packages used**

For achieving the desired output, the following packages have been imported into the program using the built-in keyword import:

* + 1. **Swing(javax.swing.\*)**

Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.

* + 1. **Abstract Windowing Toolkit(java.awt.\*)**

The Abstract Window Toolkit (AWT) is Java's original platform-dependent windowing, graphics, and user-interface widget toolkit, preceding Swing. The AWT is part of the Java Foundation Classes (JFC) — the standard API for providing a graphical user interface (GUI) for a Java program.

1. **Source Code**

“TC.java”

import javax.swing.\*;

import java.awt.event.\*;

public class TC extends JFrame

{

    JLabel l1,l2,l3,l4;

    JComboBox tc1, tc2;

    JTextField t1,t2;

    JButton b,dot,ac,bs,pm;

    JButton n0,n1,n2,n3,n4,n5,n6,n7,n8,n9;

    public TC(String s)

    {

        super(s);

    }

    public void setComp()

    {

        String arr1[] = {"Celsius","Fahrenheit","Kelvin"};

        String arr2[] = {"Celsius","Fahrenheit","Kelvin"};

        l1 = new JLabel("To");

        l2 = new JLabel("Enter the Value:");

        l3 = new JLabel("Converted Value:");

        l4 = new JLabel("From");

        tc1 = new JComboBox(arr1);

        tc2 = new JComboBox(arr2);

        t1 = new JTextField();

        t2 = new JTextField();

        b = new JButton("Convert");

        dot = new JButton(".");

        pm = new JButton("±");

        bs = new JButton("\\");

        ac = new JButton("AC");

        n0 = new JButton("0");

        n1 = new JButton("1");

        n2 = new JButton("2");

        n3 = new JButton("3");

        n4 = new JButton("4");

        n5 = new JButton("5");

        n6 = new JButton("6");

        n7 = new JButton("7");

        n8 = new JButton("8");

        n9 = new JButton("9");

        setLayout(null);

        tc1.setBounds(75,50,100,20);

        t1.setBounds(200,50,100,20);

        l1.setBounds(100,75,50,20);

        l4.setBounds(90,25,50,20);

        l2.setBounds(200,35,100,20);

        l3.setBounds(200,85,100,20);

        tc2.setBounds(75,100,100,20);

        t2.setBounds(200,100,100,20);

        b.setBounds(138,150,100,20);

        ac.setBounds(238,200,50,100);

        pm.setBounds(238,350,50,50);

        bs.setBounds(238,300,50,50);

        dot.setBounds(188,350,50,50);

        n0.setBounds(88,350,100,50);

        n1.setBounds(88,200,50,50);

        n2.setBounds(138,200,50,50);

        n3.setBounds(188,200,50,50);

        n4.setBounds(88,250,50,50);

        n5.setBounds(138,250,50,50);

        n6.setBounds(188,250,50,50);

        n7.setBounds(88,300,50,50);

        n8.setBounds(138,300,50,50);

        n9.setBounds(188,300,50,50);

        b.addActionListener(new Handler());

        ac.addActionListener(new Handler());

        pm.addActionListener(new Handler());

        bs.addActionListener(new Handler());

        dot.addActionListener(new Handler());

        n0.addActionListener(new Handler());

        n1.addActionListener(new Handler());

        n2.addActionListener(new Handler());

        n3.addActionListener(new Handler());

        n4.addActionListener(new Handler());

        n5.addActionListener(new Handler());

        n6.addActionListener(new Handler());

        n7.addActionListener(new Handler());

        n8.addActionListener(new Handler());

        n9.addActionListener(new Handler());

        add(tc1);

        add(tc2);

        add(l1);

        add(l2);

        add(l3);

        add(l4);

        add(t1);

        add(t2);

        add(b);

        add(ac);

        add(dot);

        add(pm);

        add(bs);

        add(n0);

        add(n1);

        add(n2);

        add(n3);

        add(n4);

        add(n5);

        add(n6);

        add(n7);

        add(n8);

        add(n9);

        t2.setEditable(false);

    }

    class Handler implements ActionListener

    {

        public void actionPerformed(ActionEvent e)

        {

            String x,y;

            x = (String)tc1.getSelectedItem();

            y = (String)tc2.getSelectedItem();

            if(e.getSource()==n0)

                t1.setText(t1.getText() + "0");

            if(e.getSource()==n1)

                t1.setText(t1.getText() + "1");

            if(e.getSource()==n2)

                t1.setText(t1.getText() + "2");

            if(e.getSource()==n3)

                t1.setText(t1.getText() + "3");

            if(e.getSource()==n4)

                t1.setText(t1.getText() + "4");

            if(e.getSource()==n5)

                t1.setText(t1.getText() + "5");

            if(e.getSource()==n6)

                t1.setText(t1.getText() + "6");

            if(e.getSource()==n7)

                t1.setText(t1.getText() + "7");

            if(e.getSource()==n8)

                t1.setText(t1.getText() + "8");

            if(e.getSource()==n9)

                t1.setText(t1.getText() + "9");

            if(e.getSource()==dot)

                t1.setText(t1.getText() + ".");

            if(e.getSource()==ac)

            {

                t1.setText("");

                t2.setText("");

            }

            if(e.getSource()==pm)

            {

                String spm = new String();

                spm = t1.getText();

                if(spm.length() == 0)

                    t1.setText("-");

                else if(spm.charAt(0)!='-')

                    t1.setText("-" + t1.getText());

                else

                    t1.setText("" + spm.substring(1));

            }

            if(e.getSource()==bs)

            {

                int n;

                String bsp = new String();

                bsp = t1.getText();

                n = bsp.length();

                t1.setText("" + bsp.substring(0,n-1));

            }

            if(e.getSource()==b)

            {

                //Both are same

                if(x=="Celsius" && y=="Celsius"){

                    t2.setText(""+t1.getText());

                }

                else if(x=="Fahrenheit" && y=="Fahrenheit"){

                    t2.setText(""+t1.getText());

                }

                else if(x=="Kelvin" && y=="Kelvin"){

                    t2.setText(""+t1.getText());

                }

                //Celsius to others

                else if(x=="Celsius" && y=="Fahrenheit"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)((a \* 9/5)+32);

                    t2.setText(""+b);

                }

                else if(x=="Celsius" && y=="Kelvin"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)(a + 273.15);

                    t2.setText(""+b);

                }

                //Fahrenheit to others

                else if(x=="Fahrenheit" && y=="Celsius"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)((a - 32) \* 5/9);

                    t2.setText(""+b);

                }

                else if(x=="Fahrenheit" && y=="Kelvin"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)((a - 32) \* 5/9 + 273.15);

                    t2.setText(""+b);

                }

                //Kelvin to others

                else if(x=="Kelvin" && y=="Celsius"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)(a - 273.15);

                    t2.setText(""+b);

                }

                else if(x=="Kelvin" && y=="Fahrenheit"){

                    String s = t1.getText();

                    float a = Float.parseFloat(s);

                    float b = (float)((a - 273.15) \* 9/5 + 32);

                    t2.setText(""+b);

                }

                }

            }

        }

    public static void main(String[] args)

    {

        TC jf = new TC("Temperature Converter");

        jf.setComp();

        jf.setSize(400,500);

        jf.setVisible(true);

        jf.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

    }

}

* 1. **Displayed outputs**

Graphical user interface

Description automatically generated with low confidence

* 1. **Concepts used**

These are the OOP concepts that were implemented in the source code above.

* + 1. **GUI**

GUI (Graphical User Interface) in Java is an easy-to-use visual experience builder for Java applications. It is mainly made of graphical components like buttons, labels, windows, etc. through which the user can interact with an application. GUI plays an important role to build easy interfaces for Java applications.

* + 1. **Class**

A class is a collection of objects. Unlike the primitive data structures, classes are data structures that the user defines. They make the code more manageable.

* + 1. **Objects and object instantiation**

When we define a class only the description or a blueprint of the object is created. There is no memory allocation until we create its object. The objector instance contains real data or information. Instantiation is nothing but creating a new object/instance of a class.

* + 1. **Inheritance**

Inheritance is another labor-saving Java OOP concept that works by letting a new class adopt the properties of another. We call the inheriting class a subclass or a child class. The original class is often called the parent. We use the keyword extends to define a new class that inherits properties from an old class.

* + 1. **Abstraction**

Abstraction lets programmers create useful and reusable tools. For example, a programmer can create several different types of objects, which can be variables, functions, or data structures. Programmers can also create different classes of objects as ways to define the objects.

* + 1. **Polymorphism**

Polymorphism in Java works by using a reference to a parent class to affect an object in the child class. We might create a class called “horse” by extending the “animal” class. That class might also implement the “professional racing” class. The “horse” class is “polymorphic,” since it inherits attributes of both the “animal” and “professional racing” class.

1. **Applications of the program**

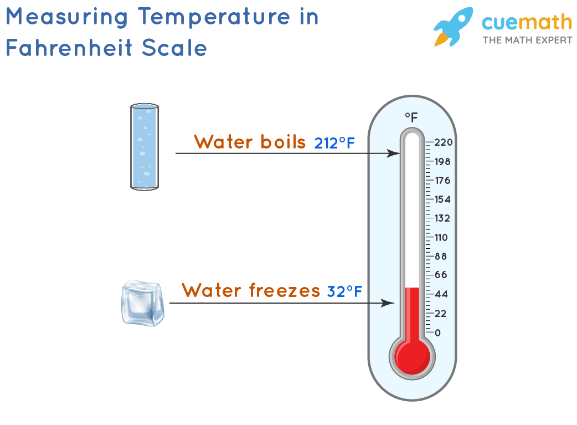
A temperature converter helps in the conversion of the measurement units of the temperature recorded in a particular unit. Temperature expresses the degree of heat or cold of a solid, liquid, or gas. Temperature is measured using a thermometer. While Kelvin (K) is the SI unit of temperature, people generally use Centigrade or Celsius (°C) and Fahrenheit (°F) to measure temperature.

For example, the lowest practical temperature on Earth is -273.15 degrees Celsius. Celsius is a measure of temperature. This recording can be represented in Kelvin as 0 K. We use temperature converter to find the different readings of a given temperature on different unit measuring scales

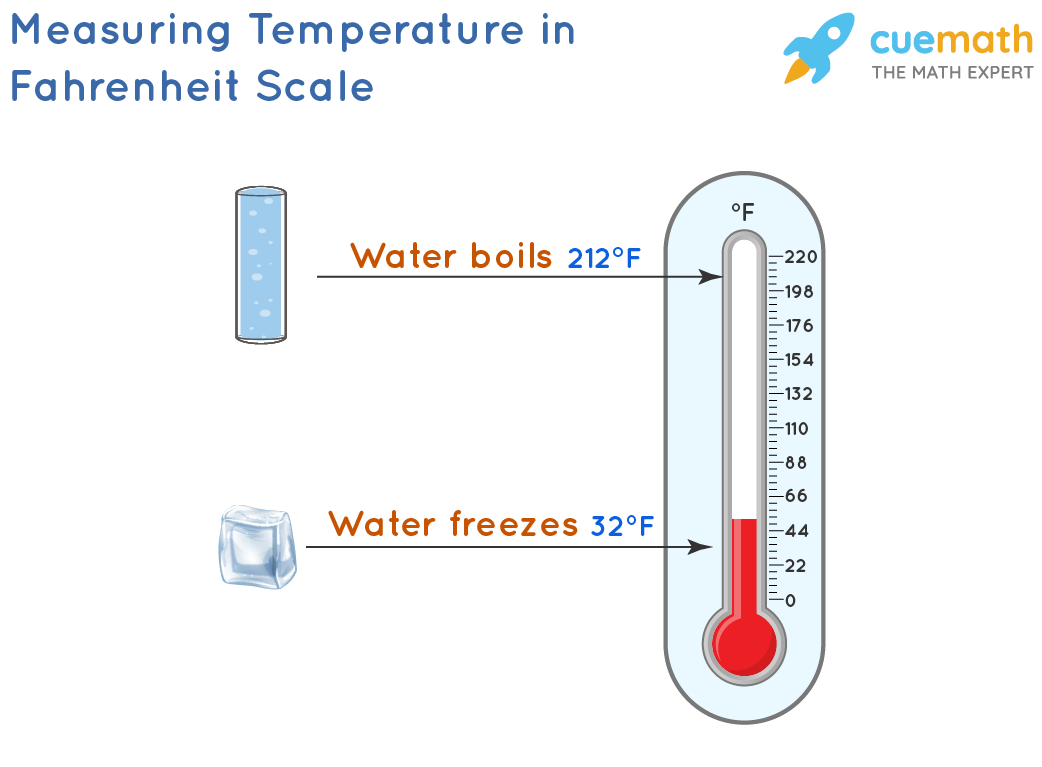
* 1. **Units of Temperature**

Different units are used to record the temperature. The three different units used for measuring temperature are Celsius (°C) Fahrenheit (°F), and Kelvin (K). Kelvin is the SI unit of measuring temperature, whereas Fahrenheit and Celsius are commonly used scales

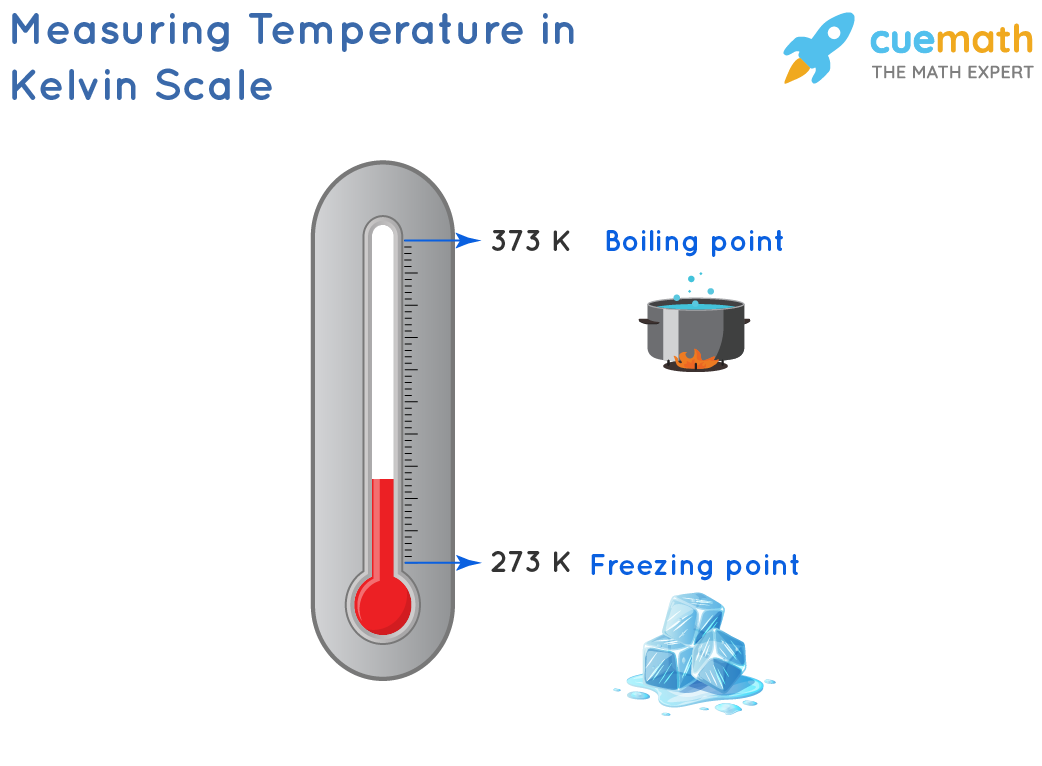
* + 1. **Celsius**

Celsius scale was Invented in 1742 by Swedish astronomer Anders Celsius and hence named after him. Celsius, also called centigrade, is based on the freezing point of water which is 0°, and the boiling point of water which is 100° The temperature in celsius is represented with °C. Normal human body temperature is 37°C.

* + 1. **Fahrenheit**

The Fahrenheit scale is a temperature scale developed by Daniel Gabriel Fahrenheit and hence named after him. This scale has the boiling point of water at 212° F and the freezing point at 32° F. The temperature in Fahrenheit is represented with °F. The normal human body temperature is 98.6°F.

* + 1. **Kelvin**

Kelvin is the SI unit of temperature. The unit symbol is K. It is named after the physicist William Thomson, 1st Baron Kelvin (1824–1907). Here the degree symbol ° is not used to represent the temperature, unlike in Celsius or Fahrenheit.

1. **Conclusion**

This article aims to show how to create an application on concepts of object-oriented programming.

**5.2 Software**

Java is very flexible - it can be used to develop software as well as applets (small programs that run on webpages). But the flexibility doesn't end there because you can run the same Java programs on various operating systems without having to rewrite the code (unlike some other languages such as C and C++) thanks to the Java run-time environment which interprets Java code and tells the operating system what to do. It also offers multiple great graphing packages that come packed with lots of different features. Here we have used awt and swing for GUI.

1. **References**

[1] “Java (Programming Language).” Wikipedia, Wikimedia Foundation, 7 Dec. 2021, <https://en.wikipedia.org/wiki/Java_(programming_language)>

[2] “Object-Oriented Programming.” Wikipedia, Wikimedia Foundation, 25 Nov. 2021,

<https://en.wikipedia.org/wiki/Object-oriented_programming>