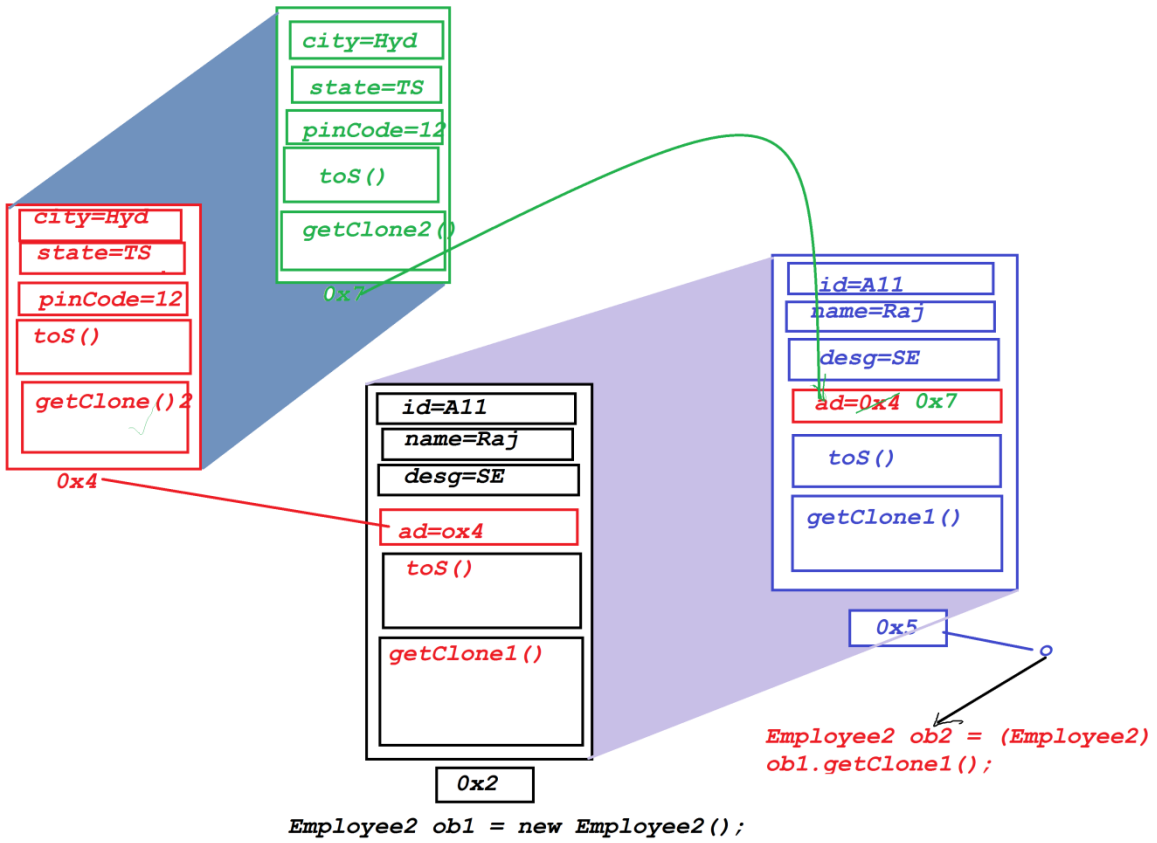
Dt : 9/11/2023

Diagram:



==========

Advantage of Cloning process:

=>Through Cloning Process we can take the backup of Objects part of Protection and Security.

==========

faq:

define Cloneable?

=>Cloneable is an empty interface from java.lang package and which specify Cloning process.

=>This Cloneable interface also known marker interface or Tagging Interface.

==========

Note:

=>Serialization process and Cloning process are Specialization processes in CoreJava.

==========

faq:

define Garbage Collection Process?

=>The process of collecting all anonymous objects and destroying is known as Garbage Collection Process.

=>Part of execution process,execution-control will run gc() method to perform garbage collection process.

=>This gc() method executed everytime after method-frame is destroyed and this gc() method will collect all anonymous objects.

=>This gc() method will internally calls finalize() method to check the objects are eligible for destroying or not,then the anonymous objects are destroyed.

=>This gc() method runs like daemon thread,which means executes contineously part of execution-process.

==========

faq:

define Anonymous Objects?

=>The Objects which are created without name are known as Anonymous Objects.

==========

faq:

wt is the diff b/w

(i)final

(ii)finally

(iii)finalize

(i)final:

=>final is a keyword used to declare variables,methods and classes.

(ii)finally:

=>finally is a block part of exception handling process and executed independently without depending on exception.

(iii)finalize:

=>finalize() is a method of Object-class used in garbage collection process to check the objects are eligible for garbage Collection

or not.

==========

faq:

define Stand-Alone Application?

=>The applications which are installed in one computer and performs actions in the same computer are known as Stand-Alone Applications or DeskTop applications or Windows Applications.

=>Based on user interaction the Stand-Alone applications are categorized into two types:

(a)CUI Applications

(b)GUI Applications

(a)CUI Applications:

=>The applications in which user interacts through Console are known as CUI Applications.(CUI - Console User Interface)

Ex:

All Above programs

(b)GUI Applications:

=>The Applications in which the user interacts through GUI Components are known as GUI Applications.(GUI - Graphical User Interface)

=>We use the following to design GUI Components:

AWT - Abstract Window Toolkit

Swing

JavaFx - Introduced by Java8 and inbuilt with CSS and UI controls

Ex-program:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;
public class Student2
extends JFrame implements ActionListener
{
        String str1,str2=null,str3,str4;
        JLabel lb1;
        JLabel lb2;
        JLabel lb3;
        JLabel lb4;
        JLabel lb5;
        JLabel lb6;
        JLabel lb7;
        JLabel lb8;
        JComboBox jc;
```

```
JTextField t1;
JTextField t2;
JTextField t3;
JTextField t4;
JTextField t5;
JTextField t6;
JButton b1;
JButton b2;

Student2() //constructor
{
        Container c=this.getContentPane();
        String str1[]=
        {"ECE","CSE","EEE","MECH","CIVIL"};
        jc= new JComboBox(str1);
        c.setLayout(null);
        c.setBackground(Color.yellow);
Font f1=new Font("dialog",Font.BOLD,30);
        lb1=new JLabel("Student Data");
        lb1.setFont(f1);
        lb1.setBounds(450,50,500,50);
        lb1.setForeground(Color.red);
```

```
Font f=new Font("dialog",Font.BOLD,20);
lb3= new JLabel("BRANCH");
        lb3.setFont(f);
        lb3.setBounds(450,100,500,50);
        lb3.setForeground(Color.red);
                jc.setFont(f);
                jc.setBounds(550,100,150,50);
                jc.setForeground(Color.GREEN);
        lb2=new JLabel("NAME");
        lb2.setFont(f);
        lb2.setBounds(50,100,500,50);
        lb2.setForeground(Color.red);
                        t1=new JTextField(50);
                t1.setBounds(200,100,200,50);
                lb4=new JLabel("RNO");
        lb4.setFont(f);
        lb4.setBounds(50,180,500,50);
        lb4.setForeground(Color.red);
                t2=new JTextField(50);
                t2.setBounds(200,180,200,50);
        lb5=new JLabel("6 SUB MARKS");
        lb5.setFont(f);
```

```
lb5.setBounds(50,260,500,50);
lb5.setForeground(Color.red);
        t3=new JTextField(50);
        t3.setBounds(200,260,300,50);
lb6=new JLabel("TOTAL");
lb6.setFont(f);
lb6.setBounds(50,340,500,50);
lb6.setForeground(Color.red);
        t4=new JTextField(50);
        t4.setBounds(200,340,150,50);
lb7=new JLabel("PERCENTAGE");
lb7.setFont(f);
lb7.setBounds(450,340,500,50);
lb7.setForeground(Color.red);
        t5=new JTextField(50);
        t5.setBounds(600,340,150,50);
        lb8=new JLabel("RESULT");
lb8.setFont(f);
lb8.setBounds(50,420,500,50);
lb8.setForeground(Color.red);
        t6=new JTextField(50);
        t6.setBounds(200,420,150,50);
```

```
        b1=new JButton("Calculate");
b1.setBounds(300,500,100,50);
        b2=new JButton("Clear");
b2.setBounds(500,500,100,50);
        c.add(lb1);
c.add(lb2);
c.add(t1);
c.add(lb3);
c.add(jc);
c.add(lb4);
c.add(t2);
c.add(lb5);
c.add(t3);
c.add(lb6);
c.add(t4);
c.add(lb7);
c.add(t5);
c.add(lb8);
c.add(t6);
c.add(b1);
c.add(b2);
b1.addActionListener(this);
```

```
        b2.addActionListener(this);
    }
    public static void main(String[] args)
    {
        Student2 obj1=new Student2();
        obj1.setTitle("Student Details");
        obj1.setSize(800,600);
        obj1.setVisible(true);
obj1.setDefaultCloseOperation
  (JFrame.EXIT_ON_CLOSE); // close window
    }
public void actionPerformed(ActionEvent arg)
    {
        str1=arg.getActionCommand();
        if(str1.equals("Calculate"))
        {
            str2=t1.getText();
            str3=t2.getText();
            try
            {
            int len=str3.length();
            if(len==10)
```

```
        {
        try
                {
String s11=str3.substring(7,8);
        Choice2 c1=new Choice2();
        String bb=c1.valid(s11);
        boolean br1=bb.equals("1");
        boolean br2=bb.equals("2");
        boolean br3=bb.equals("3");
        boolean br4=bb.equals("4");
        boolean br5=bb.equals("5");
        String ss=null;
        if(br1)
                ss="CIVIL";
        else if(br2)
                ss="EEE";
        else if(br3)
                ss="mech";
        else if(br4)
                ss="ECE";
        else if(br5)
                ss="CSE";
```

```
if(((jc.getSelectedItem().toString())
                                .equals(ss)))
                        {
                                try
                                {
                                str4=t3.getText();
StringTokenizer st=
                new StringTokenizer(str4," ");
                                int a,b,c,d,e,f;
                                String s1=st.nextToken();
                                String s2=st.nextToken();
                                String s3=st.nextToken();
                                String s4=st.nextToken();
                                String s5=st.nextToken();
                                String s6=st.nextToken();
                               a=Integer.parseInt(s1);
                                b=Integer.parseInt(s2);
                                c=Integer.parseInt(s3);
                                d=Integer.parseInt(s4);
                                e=Integer.parseInt(s5);
                                f=Integer.parseInt(s6);
            if(!((a<0 || a>100) || (b<0 || b>100) ||
```

```
                (c<0 || c>100)
        || (d<0 || d>100) || (e<0 || e>100) ||
                (f<0 || f>100)))
                                {
                        int total=a+b+c+d+e+f;
                        t4.setText(" "+total);
                        float per=total/6;
                        t5.setText(" "+per);
if((a<35 || b<35 || c<35 || d<35 || e<35 ||
                        f<35))
                                {
                        t6.setText("fail");
                                }
                                else
                                {
                        t6.setText("pass");
                                }
                        }
                        else
                        {
JOptionPane.showMessageDialog
  (this,"values between 0 to 100");
```

```
                    }
                }
            catch(NumberFormatException nfe)
                {
JOptionPane.showMessageDialog
(this,"only enter the number in marks");
                    }
                }
                else
                {
JOptionPane.showMessageDialog
(this,"mismatch of rno and branch");
                }
                }
        catch(NullPointerException npe)
                {
    JOptionPane.showMessageDialog
      (this,"invalid rno");
                }
                }
            else
                {
```

```
        JOptionPane.showMessageDialog
        (this,"rno must be 10 digits");
                        }
                }
        catch(NoSuchElementException nsee)
                {
JOptionPane.showMessageDialog
(this," plz enter 6 sub marks");
                }
        }
            else
            {
                t1.setText("");
                t2.setText("");
                t3.setText("");
                t4.setText("");
                t5.setText("");
                t6.setText("");
            }
        }
}
class Choice2
```

```
{
  String b;
  String valid(String s1)
  {
    switch(s1)
        {
          case "1":
                b="1";
                break;
          case "2":
                b="2";
                break;
          case "3":
                b="3";
                break;
         case "4":
                b="4";
                break;
         case "5":
                b="5";
                break;
        }
```

```
 return b;
  }
}
```

==========

==========

The following are the features of Java:

1.simple

2.Object-Oriented

3.Portable

4.Platform Indepedent

5.secured

6.Architecture Nutral

7.Interpreted

8.HighPerformance

9.MultThreaded

10.Distributed

11.Dynamic

12.Robust(Strong)

==========

faq:

define JShell?

=>"JShell" tool introduced by Java9 version to write

script-based code.

=>JShell is known as REPL tool

R - Read

E - Evaluate

P - Print

L - Loop

==========

Summary of CoreJava:

part-1 : CoreJava:

(1)Programming Components(Java Alphabets)

(2)Programming Concepts

(3)Object Oriented Programming features

(1)Programming Components(Java Alphabets)

(a)variables

1.Primitive datatype variables(Values)

(i)Static Variables

(ii)NonStatic Variables

=>Instance Variables

=>Local Variables

2.NonPrimitive datatype variables(Object references)

(i)Static Variables

(ii)NonStatic Variables

=>Instance Variables

=>Local Variables

(b)Methods

1.Static methods

2.NonStatic methods(Instance methods)

(c)Blocks

1.static blocks

2.NonStatic blocks(Instance blocks)

(d)Constructors

=>NonStatic Constructors

(There is no Concept of Static Constructors in Java)

(e)Classes

1.static classes(Only as InnerClasses)

2.NonStatic Classes

(f)Interfaces

1.static Interfaces(Only as InnerInterfaces)

2.NonStatic Interfaces

(g)AbstractClasses

1.static AbstractClasses(Only as InnerAbstractClasses)

2.NonStatic AbstractClasses

==========

(2)Programming Concepts

(a)Object Oriented Programming

=>The following are the levels in Object Oriented Programming:

1.Object definition - Object is a Storage related to class

2.Object Creation - we use "new" keyword to create Object

3.Object Location - Object created in Heap Area of JVM

4.Object Components - Object will hold Instance members

5.Object Types

(a)User defined Class Objects

(i)Muatble Objects

(ii)Immutable Objects

(b)String-Objects

(i)String class - Immutable Objects

(ii)StringBuffer class - Mutable Objects - Synchronized class

(iii)StringBuilder class - Mutable Object - NonSynchronized Class

=>Utility Classes

(i)StringTokenizer class

(ii)StringJoiner class

(c)WrapperClass Objects

(i)Byte Object

(ii)Short Object

(iii)Integer Object

(iv)Long Object

(v)Float Object

(vi)Double Object

(vii)Character Object

(viii)Boolean Object

(d)Array Objects

(i)Array holding User defined Class Objects

(ii)Array holding WrapperClass Objects

(iii)Array holding String-Objects

(iv)Array holding DisSimiler Objects(Object-Array)

(v)Array holding Array-Objects(Jagged Array)

(e)Collection<E> Objects

1.Set<E>

(i)HashSet<E>

(ii)LinkedHashSet<E>

(iii)TreeSet<E>

2.List<E>

(i)ArrayList<E>

(ii)LinkedList<E>

(iii)Vector<E>

=>Stack<E>

3.Queue<E>

(i)PriorityQueue<E>

=>Deque<E>

(ii)ArrayDeque<E>

(iii)LinkedList<E>

(f)Map<K,V> Objects

(i)HashMap<K,V>

(ii)LinkedHashMap<K,V>

(iii)TreeMap<K,V>

(iv)Hashtable<K,V>

(g)Enum<E> Objects

6.Objects Collection(Grouping Objects)

7.Objects Sorting

8.Object Locking

9.Object Serialization

10.Object Cloning

==========

****imp***

=>Relations in Java

(i)References

(ii)Inheritance

(iii)InnerClasses/InnerInterfaces/InnerAbstractClases

==========

(b)Exception Handling process

1.Exception definition

2.Exception Vs Error

3.Exception Handling process

(i)try

(ii)catch

(iii)finally

4.Hierarchy of "Throwable"

5.Types of Exceptions

(i)Checked Exceptions

(ii)UnChecked Exceptions

6.Exception re-throwing process

7.Exception Propagation

8.try-with-resource statement(Java7)

9.Enhanced try-with-resource statement(Java9)

10."java.lang.NullPointerException"

==========

(c)Java Collection Framework(JCF)

(Data Structures in Java)

1.Array

2.Collection<E>

(i)Set<E>

(ii)List<E>

(iii)Queue<E>

3.Map<K,V>

(d)Multi-Threading Concept

=>Thread definition

=>Thread creation

=>Thread Location

=>Thread Synchronization

(1)Mutual Exclusion

(i)Synchronized block

(ii)Synchronized method

(iii)static Synchronization

(2)Thread Communication

=>wait() Vs notify() Vs notifyAll()

=>Thread Life-Cycle

(e)File Storage

=>Stream

=>Types of Streams

(i)Byte Stream

(ii)Character Stream

=>FileInputStream Vs FileOutputStream

=>FileReader Vs FileWriter

=>Object Serialization

(i)Serialization

(ii)DeSerialization

=>ObjectInputStream Vs ObjectOutputStream

=>readObject() Vs writeObject()

=>java.io.Serializable interface

=>java.io.File class

(f)Networking in Java

=>Network definition

=>Types of Networks

=>Server Computer Vs Client Computer

=>Network Protocols

=>IP Address

=>Socket Vs PortNo

=>Socket Vs ServerSocket

==========

(3)Object Oriented Programming features:

(a)Class

=>Class is a 'structured layout' generating Objects.

=>Based on Security:

(i)Mutable Classes

(ii)Immutable classes

(b)Object

=>Object is a storage related to a class holding Instance members of class.

=>Based on Security,Objects are categorized into two types:

(i)Mutable Objects

(ii)Immutable Objects

(c)Abstraction

=>The process of hiding the background implementation which is not needed by the EndUsers is known as 'Abstraction process'

=>we use "Interfaces" and "AbstractClasses" to perform abstraction process.

(d)Encapsulation

=>The process of binding all the programming components into a single unit class is known as "Encapsulation process".

=>Which means,Class is holding variables,methods,blocks,Constructors, InnerClases,InnerInterfaces,InnerAbstractClasses and Exception handling components like try,catch,finally,throw and throws.

(e)PolyMorphism

=>The process in which programming Components having more than one form is known as PolyMorphism.

(i)Dynamic PolyMorphism

=>Method Overriding process

(ii)Static PolyMorphism

=>Method Overloading process

=>Objects will have two forms:

(i)Mutable Objects

(ii)Immutable Objects

(f)Inheritance

=>The process of interlinking classes with "extends" keyword is known as Inheritance.

=>Types of Inheritances:

1.Single Inheritance

2.Multiple Inheritance

3.Multi-Level Inheritance

4.Hirarchal Inheritance

5.Hybrid Inheritance

=>According to realtime application development the Inheritances are categorized into two types:

1.Single Inheritance

2.Multiple Inheritance(Using Interfaces)

---

faq:

define data structure?

=>data structure is a defined format organizing,processing,retrieving and storing data.

=>data structure will hold well organized data.

=>According to Java,the following are data structure Components:

1.Array

2.Collection<E>

(i)Set<E>

(ii)List<E>

(iii)Queue<E>

3.Map<K,V>

==========