Unit 4 – ICS3U0 – Java Applets

Sample Test - April 26, 2024

Name: Gorski

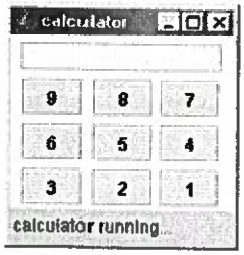
Total	%	Knowledge	Communication	Thinking	Application
(80)	%	(20)	(20)	(20)	(20)

Communication

* one of the list is fine.

1. Outline one way to achieve the indicated aspect of user-interface design in the applet shown. /3

(a)	Visually appealing	- use colour - change font not: similar widgets format similarly. - use pictures not: resize Applet
(b)	Restricts Input	- use JLabel instead of JTextField on display - Disable Buttons when they shouldn't be clicked
(c)	Error Handling	- add a reset/undo button - add ifs to handle invalid data entry



2. Which colour is created by each line of code? /6

(a)	`new Color(0,255,0)` (R G B)	green	(d)	`new Color(0,255,0)`	green ☹ bad proof-reading
(b)	`new Color(255,0,255)`	magenta	(e)	`new Color(255,255,0)`	yellow
(c)	`new Color(0,0,0)`	black	(f)	`new Color(255,255,255)`	white

3. Fill in the following about applets. /8

new	(a) What is the word that signals construction?
get (write (g) clearly)	(b) What is the word that signals an accessor? mutator = set accessor – seeing what is in memory
declare	(c) What is the term for allocating memory for a variable?
construction	(d) What is the term for setting up RAM for a widget to use?
mutation	(e) What is the term for changing a widget's RAM?
command line interface	(f) What does CLI abbreviate? GUI = Graphic User Interface
1	(g) How many times does init run? when? at start
many (as many as user wants)	(h) How many times does ActionPerformed run? when? when button clicked

4. Why is user-centric design important? Use a specific example. (3 sentences) /3

User centric design uses if-statements to guard against mistakes. Preventing errors is important because everyone has "off-days" and incorrect data-entry can cause serious problems. For example, a tired, over-worked nurse accidentally entered 4 days of medication in 4 hours on Denise Melanson's chemo pump with tragic results. An if in the code would have prevented it.

Knowledge

5. Look at the code below.
Circle the correct answer: /10

T [✓ F] a. You should save this applet as Calculator.java.

T [✓ F] b. There is a `createImageIcon` method in this program.

T [✓ F] c. An `int` is a widget.

[✓ T] F d. An `accessor` used in the program is `getText`.

T [✓ F] e. There are 7 widgets on the screen.

[✓ T] F f. `setText` is a mutator method.

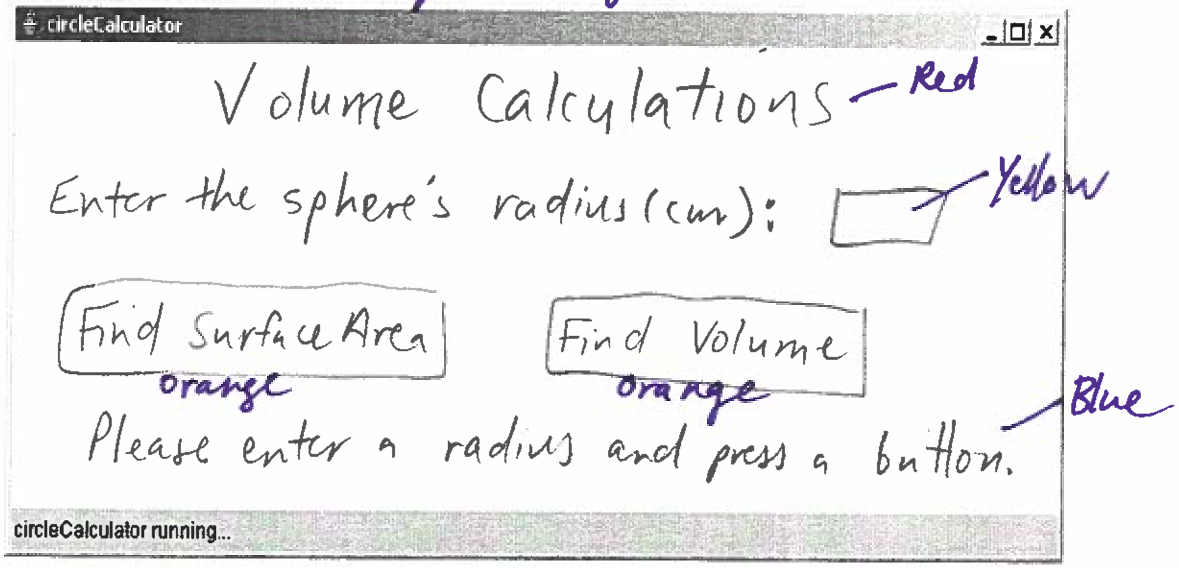
[✓ T] F g. The `output` widget is globally declared.

T [✓ F] h. The widget whose text is changed in `actionPerformed` is `showStatus`.

T [✓ F] i. There are 3 actionCommands.

[✓ T] F j. The line `add(title)` can be used in init, but can not be used in ActionPerformed.

6. Fill in the applet's screen using the code shown below. /10
Be careful to **label the colours.** * Don't forget to label colours *


circleCalculator
Volume Calculations — Red
Enter the sphere's radius (cm):
Yellow
Find Surface Area
Orange
Find Volume
Orange
Please enter a radius and press a button.
Blue
circleCalculator running...

```
import java.applet.*; import java.awt.*; import java.awt.event.*; import javax.swing.*;

public class circleCalculator extends Applet implements ActionListener {
    JTextField radius;
    JLabel output;

    public void init ()
    {
        JButton b1 = new JButton ("Find Surface Area");
        b1.addActionListener (this);
        b1.setActionCommand ("calc1");
        b1.setBackground(Color.orange);

        JButton b2 = new JButton ("Find Volume");
        b2.addActionListener (this);
        b2.setActionCommand ("calc2");
        b2.setBackground(Color.orange);

        JLabel title = new JLabel ("Volume Calculations");
        title.setFont (new Font ("Ravie", Font.PLAIN, 20));
        title.setForeground (Color.red);

        JLabel n = new JLabel ("Enter the sphere's radius (cm):");

        radius = new JTextField (6);
        radius.setBackground (Color.yellow);

        output = new JLabel ("Please enter a radius and press a button.");
        output.setForeground (Color.blue);
        add (title);     1
        add (n);         2
        add (radius);    3
        add (b1);        4
        add (b2);        5
        add (output);    6
    }

    public void actionPerformed (ActionEvent e)
    {
        int r = Integer.parseInt (radius.getText ());
        if (r <= 0)
            output.setText ("Enter a positive radius.");
        else if (r > 10000)
            output.setText ("Enter a smaller number.");
        else if (e.getActionCommand ().equals ("calc2")) {
            double v = 4.0 / 3.0 * Math.PI * r * r * r;
            output.setText ("The volume is " + v + ".");
        }
        else if (e.getActionCommand ().equals ("calc1")) {
            double sa = 4.0 * Math.PI * r * r;
            output.setText ("The surface area is " + sa + ".");
        }
        showStatus ("Thank you.");
    }
}
```

Add order here
6 widgets added

7. Code the init method of the Tulip applet on the right. /10

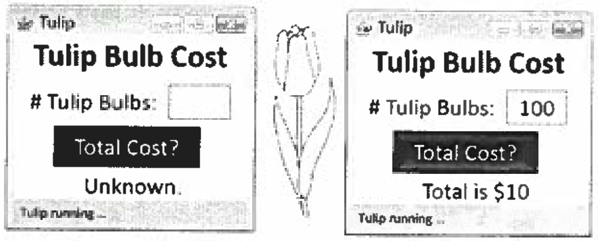
```
import java.awt.*; import javax.swing.*; import java.applet.Applet; import java.awt.event.*;
public class Tulip extends Applet implements ActionListener
{
    JTextField num;
//The other global variable:
    JLabel  output ;
    public void init ()
    {  resize (300, 100);
//The first label: (Font is Arial, Font.BOLD and 30 pt)
    JLabel title = new JLabel ("TulipBulbCost");
    title.setFont(new Font("Arial", Font.BOLD,30));
//The prompt
    JLabel pmt = new JLabel ("#Tulip Bulbs");
//The textfield:
    num = new JTextField (4);
//The button: (Black background, white writing)
    JButton b = new JButton ("Total Cost?");
    b.setBackground(Color.black);
    b.setForeground(Color.white);
    b.addActionListener(this);
    b.setActionCommand("whatever");
//The last label:
    output = new JLabel("Unknown");
//add the widgets:
    add(title);
    add(pmt);
    add(num);
    add(b);
    add(output);
} //init
```

= must match

— must match

Application

8. Code the Tulip applet's actionPerformed method.



Tulip

Tulip Bulb Cost

Tulip Bulbs:

Total Cost?

Unknown.

Tulip running ...

Tulip

Tulip Bulb Cost

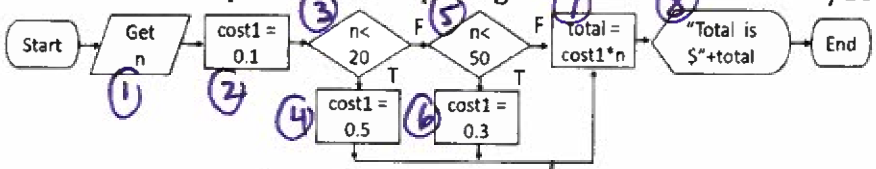
Tulip Bulbs: 100

Total Cost?

Total is $10

Tulip running ...

This flow chart explains the bulb pricing: /10



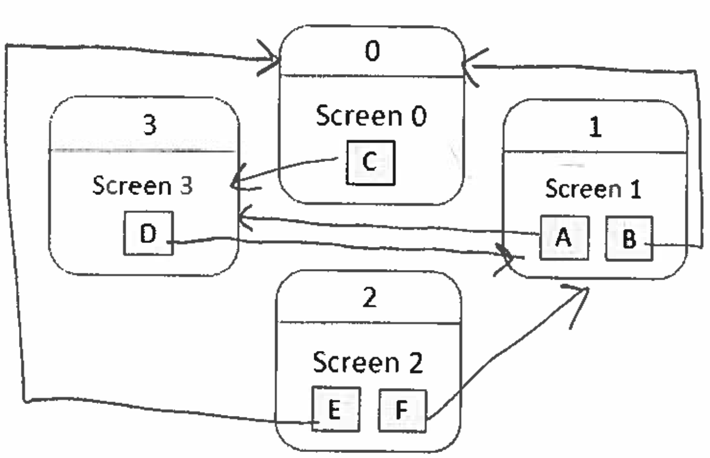
```
public void actionPerformed (ActionEvent e)
{
(1)  int n = Integer.parseInt( num .getText());
(2)  double cost1 = 0.1;
(3)  if (n<20)
(4)       cost1 = 0.5;
(5)  else if (n < 50)
(6)       cost1 = 0.3;
(7)  double total = cost1 * n;
(8)  output.setText ("Total is $ "+total);
 } //actionPerformed
} //Applet
```

Thinking

9. Circle the location where each line of code should be placed. /6

a)	import java.applet.Applet;	[✓ libraries]	global	init	actionPerformed
b)	add(b1);	libraries	global	[✓ init]	actionPerformed
c)	JLabel result;	libraries	[✓ global]	init	actionPerformed
d)	result.setText ("Diameter = " + (r * 2));	libraries	global	init	[✓ actionPerformed]
e)	radius = new JTextField (6);	libraries	global	[✓ init]	actionPerformed
f)	int r = Integer.parseInt (radius.getText ());	libraries	global	init	[✓ actionPerformed]

10. Fill in the arrows on the screen flow diagram using the actionPerformed method. Lines should not cross. Lines should not have curves, only line segments. /5



```
public void actionPerformed (ActionEvent e)
{ //moves between the screens
  if (e.getActionCommand ().equals ("A"))
      cdLayout.show (p_card, "3");
  else if (e.getActionCommand ().equals ("B"))
      cdLayout.show (p_card, "0");
  else if (e.getActionCommand ().equals ("C"))
      cdLayout.show (p_card, "3");
  else if (e.getActionCommand ().equals ("D"))
      cdLayout.show (p_card, "1");
  else if (e.getActionCommand ().equals ("E"))
      cdLayout.show (p_card, "0");
  else if (e.getActionCommand ().equals ("F"))
      cdLayout.show (p_card, "1");
}
```

11. Circle **and correct** 5 errors in this button code. /5

JButton

JButton start = new JLabel("Enter");

start.addActionListener(this);

start.~~add~~ set ActionCommand("Enter");

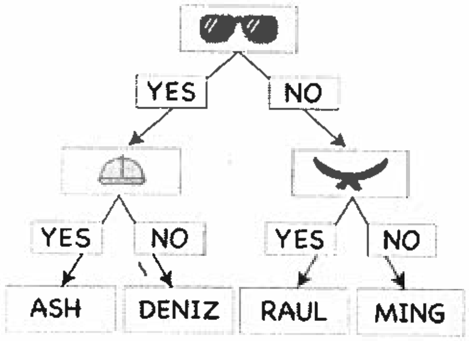
start.setPreferredSize (new Dimension (250, 20)) ;

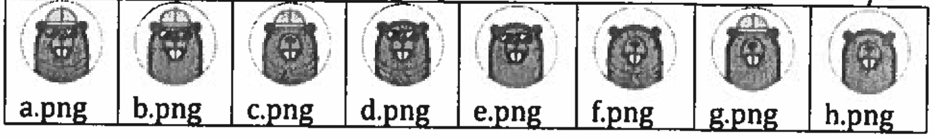
start.setBackground(Color.~~black~~ white); (can't read black on black)

start.setForeground(Color.black);

add(~~JButton~~ start);

12. The beavers are named using the decision tree on the left. Put the right picture in each button. /4





JButton **Ash** = new JButton(createImageIcon(b .png));

JButton **Deniz** = new JButton(createImageIcon(e .png));

JButton **Raul** = new JButton(createImageIcon(f .png));

JButton **Ming** = new JButton(createImageIcon(h .png));

others were possible
above were most correct.