Content Objective: Students will begin to work with abstract classes and methods while extending interfaces in Java.

|  |  |
| --- | --- |
| **On the Tech Horizon (10pts.)**  **link to a tech/coding related article or journal no more than one month old (no blogs or reddit clones see below)** | |
| URL: |  |
| Reaction/Commentary: |  |

|  |  |
| --- | --- |
| **Tech Terms and History (20pts.)**  **vocabulary from BJ p.463-512 and The Information Chapter 8 (definition/commentary/significance in your words)** | |
| Syntax to create interface | public interface InterfaceName{  //name of method  } |
| Syntax to implement interface | public class OtherClass implements InterfaceName{  public double nameOfMethod(){  return balance;  }  } |
| Why can an interface NOT be instantiated? | Interfaces are not classes, they don’t have objects that can be instantiated |
| Syntax for constants in Java | All variables in an interface have the public static final modifiers automatically  int final constant = 5;  Constants should be all caps with underscores for spaces |
| Syntax to convert class to interface (2 ways) | Declare an object of the class with the type of the interface (Measurable max = new Measurable[3] when max is a Person) |
| Syntax to cast interface to class | Country maxCountry = **(Country)** max; //The country is the cast to change max into a Country  String name = maxCountry.getName(); |
| Syntax for clone Method | BankAccount clonedAccount = (BankAccount) account.clone(); |
| Syntax for cloneable Interface | public class Object {  protected Object clone() throws CloneNotSupportedException {  if (this instanceof Cloneable) {  // Copy the instance variables  } else {  throw new CloneNotSupportedException();  }  }  } |
| What is a Callback | Executable code which is passed as an argument |
| What is an “inner class” and when/how would you use it? | It’s basically an inception class, a class which extends a subclass of another class. Ised when specific methods need to be implemented |
| What is an “anonymous class” and when/how would you use it? | Country belgium = new Country(“Belgium”, 30510);  countries.add(belgium);  Local class without a name, basically |
| What is a “mock object” and when/how would you use it? | Where you create an object without creating a fully functional object class.  //Original code  public void addScore(int studenId, double score)  public double getAverageScore(int studentId)  public void save(String filename)  //interface used  public interface IGradeBook {  void addScore(int studenId, double score);  double getAverageScore(int studentId);  void save(String filename);    } |
| What is an event listener? | Describes input actions |
| Syntax for creating an inner class listener | JButton button = new JButton(“. . .”);  // This inner class is declared in the same method as the button variable  class MyListener implements ActionListener {  . . .  };  Action Listener listener = new MyListener();  button.addActionListener(listener); |
| Syntax for mousePress() | class MousePressListener implements MouseListener {  public void mousePressed(MouseEvent event) {  int x - event.getX();  int y = event.getY();  component.moveRectangleTo(x, y);  }    // Do-nothing methods  public void mouseReleased(MouseEvent event) {}  public void mouseClicked(MouseEvent event) {}  public void mouseEntered(MouseEvent event) {}  public void mouseExited(MouseEvent event) {}  } |
| Summarize Turing’s downfall in 50-100 words. | Turing created a supercomputer that was out of its time. The computer was designed to decode enigma (which it did). He was accused of being homosexual, which was frowned upon at the time. He was given estrogen shots to humiliate him. He soon committed suicide. |
| Summarize Shannon’s advancements through the 50’s and 60’s in 50-100 words. | TI p.259-268 |

|  |  |
| --- | --- |
| **Code Snippets (30pts.)**  **only submit snippets or classes no full programs required (test and run in IDE, then copy/paste applicable code frag)** | |
| E10.3-10.4 | E10.3  public class Person implements Measurable{  private String name;  private double height;    public Person(String name, double height){  this.name = name;  this.height = height;  }    public double getHeight(){  return height;  }    public String getName(){  return name;  }      //Must have getMeasure to compile if you're implementing Measurable  public double getMeasure(){  return height;  }  }  E10.4  public static Measurable max(Measurable[] objects){  if(objects.length==0){  return null;  }  Measurable max = objects[0];  for(Measurable obj: objects){  if(obj.getMeasure() > max.getMeasure()){  max = obj;  }  }  return max;  } |
| E10.9 | class Bag {  int count;    private ArrayList<Item> bag = new ArrayList<Item>();    public void add(String itemName){  bag.add(itemName);  }    public int count(String itemName){  count = 0;  for(int i = 0, i< bag.size(),i++){  if(bag.get(i) == itemName){  count++  }  }  return count;  }  }  class Item {  String name;  int quantitiy;    public Item(String tempName, int tempQuantity){  name = tempName;  quantity = tempQuantity;  }  } |
| E10.10-E10.11 | E10.10  class Grid {      public Grid(int numRows, int numColumns){  String[][] describers = new String[numRows][numColumns];  }  public void add(int row, int column, String description){  describers[row][column] = description;  }  public String getDescription(int row ,int column){  return describers[row][column];  }  public ArrayList<Location> getDescribedLocations(){  }  }  E10.11  class Grid {    String stuff;    public Grid(){  ArrayList<String> x = new ArrayList<String>();  ArrayList<String> y = new ArrayList<String>();  }  public void add(String description){  x.add(description);  y.add(description);  }  public String getDescription(int index){  return x.get(index);  return y.get(index);  }  public ArrayList<Location> getDescribedLocations(){  for(int i = 0; i < x.size();i++){  for(int j = 0; j < y.size();j++){  if(describers[i][j] != ""){  stuff += describers[i][j] + " ";  }  }  }  }  } |
| E10.13-E10.17 | E10.13  public class ClickListener implements ActionListener  {  int n = 0;    public void actionPerformed(ActionEvent event)  {  if (true) {  n++;  }    if (n==1) {  System.out.println("I was clicked " + n + " time");  } else {  System.out.println("I was clicked " + n + " times");  }  }  }  E10.14  public class ButtonViewer  {  private static final int FRAME\_WIDTH = 300;  private static final int FRAME\_HEIGHT = 100;  public static void main(String[] args)  {  JFrame frame = new JFrame();  JButton button1 = new JButton("Button 1");  button1.setSize(100,100);  JButton button2 = new JButton("Button 2");  button2.setSize(100,100);  frame.add(button1);  frame.add(button2);    ActionListener listener1 = new ClickListener();  ActionListener listener2 = new ClickListener();  button1.addActionListener(listener1);  button2.addActionListener(listener2);  frame.setSize(FRAME\_WIDTH, FRAME\_HEIGHT);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  frame.setVisible(true);  }  }  E10.15+16 combined  public void actionPerformed(ActionEvent event)  {  if (true) {  n++;  }    if (n==1) {  System.out.println(event.getActionCommand() + " was clicked " + n + " time");  } else {  System.out.println(event.getActionCommand() + " was clicked " + n + " times");  }  }  E10.17  public void actionPerformed(ActionEvent event)  {  Date time = new Date();  String dateString = time.toString();    if (true) {  n++;  }    if (n==1) {  System.out.println(event.getActionCommand() + " was clicked " + n + " time at " + time);  } else {  System.out.println(event.getActionCommand() + " was clicked " + n + " times at " + time);  }  } |
| E10.21-E10.22 | E10.21  private boolean right = true;  private boolean down = true;  public void moveRectangleBy(int dx, int dy)  {  if(BOX\_X == 0){  right = true;  }  if(BOX\_X == 300-BOX\_WIDTH){  right = false;  }  if(BOX\_Y == 0){  down = true;  }  if(BOX\_Y == 400 - BOX\_HEIGHT){  down = false;  }  if(right && down){  box.translate(dx, dy);  }  if(right && !down){  box.translate(dx, -dy);  }  if(!right && down){  box.translate(-dx, dy);  }  if(!right && !down){  box.translate(-dx, -dy);  }  repaint();  }  }  E10.22  import java.awt.Graphics;  import java.awt.Graphics2D;  import java.awt.Rectangle;  import javax.swing.JComponent;  import java.util.\*;  /\*\*  This component displays a rectangle that can be moved.  \*/  public class RectangleComponent2 extends JComponent  {  private static final int BOX\_X = 100;  private static final int BOX\_Y = 100;  private static final int BOX\_WIDTH = 20;  private static final int BOX\_HEIGHT = 30;  private Rectangle box;  private ArrayList<Rectangle> boxes;    public void createRectangle(int mousex, int mousey){    boxes.add(new Rectangle(mousex, mousey ,BOX\_WIDTH, BOX\_HEIGHT));  repaint();  }  public RectangleComponent2()  {  boxes = new ArrayList<Rectangle>();    }  public void paintComponent(Graphics g)  {  Graphics2D g2 = (Graphics2D) g;    if(boxes.size()>0){  for(int i = 0; i < boxes.size(); i++){  g2.draw(boxes.get(i));  }  }  }  /\*\*  Moves the rectangle to the given location.  @param x the x-position of the new location  @param y the y-position of the new location  \*/  public void moveRectangleTo(int x, int y)  {  box.setLocation(x, y);  repaint();  }  }  import java.awt.event.MouseListener;  import java.awt.event.MouseEvent;  import javax.swing.JFrame;  /\*\*  This frame contains a moving rectangle.  \*/  public class RectangleFrame2 extends JFrame  {  private static final int FRAME\_WIDTH = 300;  private static final int FRAME\_HEIGHT = 400;  private RectangleComponent2 scene;  class MousePressListener implements MouseListener  {  public void mousePressed(MouseEvent event)  {  int x = event.getX();  int y = event.getY();  //scene.moveRectangleTo(x, y);  scene.createRectangle(x, y);  }  // Do-nothing methods  public void mouseReleased(MouseEvent event) {}  public void mouseClicked(MouseEvent event) {}  public void mouseEntered(MouseEvent event) {}  public void mouseExited(MouseEvent event) {}  }    public RectangleFrame2()  {  scene = new RectangleComponent2();  add(scene);  MouseListener listener = new MousePressListener();  scene.addMouseListener(listener);  setSize(FRAME\_WIDTH, FRAME\_HEIGHT);  }  } |

|  |  |
| --- | --- |
| **Code Challenge (30pts.)**  **full functioning application sent to** [**GitHub**](https://github.com/SkylineHigh/CSAdvanced/tree/master/06%20Abstract%20Data%20Types) | |
| You may choose any one of the following code challenges: P10.8-P10.21 and post code to GitHub under “Abstract Data Types” | |
| Notes: | [https://github.com/SkylineHigh/CSAdvanced/tree/master/10 Overriding/MathewBeseris](https://github.com/SkylineHigh/CSAdvanced/tree/master/10%20Overriding/MathewBeseris) |

|  |  |
| --- | --- |
| **Badge Progress (10pts.)**  **building your coding profile: Java coding training site to earn badges (recommended site** [**http://coderbyte.com**](http://coderbyte.com) **)** | |
| Screenshot/URL: |  |
| Notes/Issues: |  |

|  |  |
| --- | --- |
| **Notes**  **your notes** | |
| Notes: |  |