Content Objective: Students will implement their own classes for Java applications.

|  |  |
| --- | --- |
| **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.81-p.130 and The Information Chapter 2 (definition/commentary/significance in your words)** | |
| Instance vs. Local Variable | Local variables are declared inside of methods, instance variables are where objects store variables. (in the instance of the class) |
| Access Specifier | Declares what can access a variable or class |
| Private, Public, Protected | Public- any class can access it; Private-Only things that are able to access methods and variables are able to; Protected-the class itself, classes in the same package and subclasses can access |
| Member Method | Methods inside of a specific class |
| Encapsulation | Having hidden implementations but being publicly published |
| Public Interface | The information of a class given to people. Operations anyone can use to manipulate the class. |
| Private Implementation | Hiding or making variables only accessible by methods in the class |
| Constructor | The blueprint used to initialize different instantiations of an object |
| Parameter vs. Argument | Parameters are used to input data into a method, arguments are things executed inside the method |
| Unit Test | Testing all functions in a class outside of the program to see if it is functioning properly |
| Garbage Collector | The JVM takes back objects when they aren’t used anymore |
| Local Variable | Declared in the body of a method |
| this | Used to access implicit parameter of object it is used on |
| Syllabary | Phonetic writing system- each character represents a syllable |
| Memory was aided by? | Meter, the formulaic redundancy and poetry |
| Abstract Thought | Body, space, matter, proportion, permanence, change, quality, quantity |
| Classifying/Categorization | Aristotle tried to classify or systemize knowledge |
| Abstraction to Logic | Logic turns abstract things into more reasonable information, quantifiable |
| “To go by your words…” | Talking about how people who can write tend to lose some extent of originality in thought |
| Horses and Classes | The white horse isn’t a horse because they are 2 different things, the horse is a horse but its not white |
| Paradox and Mathematics | Do paradoxes come from language or do they exist everywhere…even in math |
| What is base 60? | Sexagesimal- the first 60 digits have their own character |
| Early Algorithm | A “procedure” for obtaining the same number of something |
| Algorithm and Procedure | A procedure as compared to a stack program, would be a list of instructions |
| Thinking and Writing | Writing is really just silent speaking. Our minds will create auditory code when we look at words, hence why we have a little voice in our head. |

|  |  |
| --- | --- |
| **Code Snippets (30pts.)**  **only submit snippets or classes no full programs required (test and run in IDE, then copy/paste applicable code frag)** | |
| R3.16 & R3.17 | BJ p.102-3 for checklists  R3.16  Counter c1 = new Counter();  System.out.println(c1.getValue());  c1.click();  System.out.println(c1.getValue());  c1.reset();  System.out.println(c1.getValue());  R3.17  Car myCar = new Car(70);  myCar.addGas(40);  myCar.drive(100);  myCar.addGas(90);  System.out.println("Expected: 100 Actual: " + myCar.getGasInTank()); |
| R3.22 & R3.24 | BJ p.112-115  R3.22  You would need a House class that has the instructions for drawing a house and a HouseComponent to actually make it draw, and to add the house component to the frame  R3.24  Put a parameter in the car instantiation and let values be inputted that change the width and height of the car |
| E3.3-E3.8 | E3.3  BankAccount account1 = new BankAccount();  BankAccount account2 = new BankAccount(500);    account1.deposit(20);  System.out.println("Balance predicted for account 1: 20 Actual Balance: " +account1.getBalance());  account2.withdraw(200);  System.out.println("Balance predicted for account 2: 300 Actual Balance: " +account2.getBalance());  E3.4  public class TestClass{  public static void main(String[] args){  BankAccount myMoney = new BankAccount(1000);  myMoney.addInterest(10);  System.out.println("Expected: 1100 Actual: "+ myMoney.getBalance());  }  }  public void addInterest(double rate){  balance = balance + rate\*.01\*balance;  }  E3.5  public class SavingsAccountTester{  public static void main(String[] args){  SavingsAccount acc1 = new SavingsAccount(1000,10);  acc1.addInterest();    System.out.println("Expected: 1100 Actual:" + acc1.getBalance());  }  }  public class SavingsAccount{  private double balance;  private double interest;    public SavingsAccount(){  balance = 0;  interest = 5;  }    public SavingsAccount(double initialBalance,double initialInterest){  balance = initialBalance;  interest = initialInterest;  }    public void addInterest(){  balance = balance + interest\*.01\*balance;  }    public double getBalance(){  return balance;  }  }  E3.6  public String printReceipt(){  String r = "Prices of items purchased: " + receipt + " Total price: " + receiptPrice;  return(r);  }  /\*\*  Records the sale of a tax-free item.  @param amount the price of the item  \*/  public void recordPurchase(double amount)  {  purchase = purchase + amount;  receipt.concat(String.valueOf(amount));  receipt.concat(" ");  receiptPrice += amount;  }  E3.7  public void reset(){  salesCount = 0;  receiptPrice = 0;  }    public int salesCount(){  return salesCount;  }    public double getSalesTotal(){  return receiptPrice;  }  public void recordPurchase(double amount)  {  purchase = purchase + amount;  receipt.concat(String.valueOf(amount));  receipt.concat(" ");  receiptPrice += amount;  salesCount++;  }  E3.8  public class EmployeeTester{  public static void main(String[] args){  Employee employee1 = new Employee("Jared", 2000);    System.out.println(employee1.getName());  System.out.println(employee1.getSalary());  employee1.raiseSalary(20);  System.out.println(employee1.getSalary());  }  }  public class Employee{    private String name;  private double salary;      public Employee(String employeeName,double currentSalary){  name = employeeName;  salary = currentSalary;  }    public String getName(){  return name;  }    public double getSalary(){  return salary;  }    public void raiseSalary(double percent){  salary = salary + salary\*percent\*.01;  }  }  BJ p.93-103 |
| E3.12 | BJ p.125  public class BugTester{  public static void main(String[] args){  Bug bug1 = new Bug(5);  bug1.move();  bug1.move();  bug1.move();  bug1.move();  bug1.turn();  bug1.move();  bug1.move();  bug1.turn();  bug1.move();  System.out.println("Expected: 8 Actual: " + bug1.getPosition());  }  }  public class Bug{    int position;  int direction;      public Bug(int position){  this.position = position;  direction = 1;  }        public void turn(){  direction \*= -1;  }    public void move(){  position += direction;  }    public int getPosition(){  return position;  }  } |

|  |  |
| --- | --- |
| **Code Challenge (30pts.)**  **full functioning application sent in a Zip file to Canvas** | |
| Using programming projects from chapter 3 (BJ p.127-128), students may choose **one** of the following challenges P3.4, P3.5, P3.7, P3.8 or P3.9. As a part of this assignment, students will need to find a partner to independently write their own unit test for the class that is implemented. You may use Canvas to send a zip file to another student. The final submission will need to include all applicable Java source code files and a unit test file from a peer (containing their name in a comment or within the file name), compressed into one .zip file for grading. | |
| Zip File Contains: | RoachSimulation.java (Entry point)  RoachPopulation.java |
| Notes: |  |

|  |  |
| --- | --- |
| **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: |  |