Name:

Course & Year: BEET 3C

Date Submitted: April 4, 2022

Post-Test Direction: In one paragraph, answer each of the following questions below. 5 points each question.

Questions:

1. Distinguish procedural and oriented in terms of programming methods

- In procedural programming, program is divided into small parts called functions. In object-oriented programming, program is divided into small parts called objects.

1. State the purpose of object orientation

- the concept of classes and objects. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects.

1. Explain the representation of object oriented and the four pillars of object-oriented programming

- Object-oriented programming is based on the concept of objects. The four pillars for OOP are Abstraction, Encapsulation, Inheritance, Polymorphism.

1. Evaluate encapsulation, abstraction, inheritance and polymorphism based on the principles of object-oriented programming.

* Abstraction: Abstraction is the process of showing only essential/necessary features of an entity/object to the outside world and hide the other irrelevant information. For example, to open your TV we only have a power button, It is not required to understand how infra-red waves are getting generated in TV remote control.
* Encapsulation: Encapsulation means wrapping up data and member function (Method) together into a single unit class. Encapsulation automatically achieve the concept of data hiding providing security to data by making the variable as private and expose the property to access the private data which would be public.
* Inheritance: The ability of creating a new class from an existing class. Inheritance is when an object acquires the property of another object. Inheritance allows a class (subclass) to acquire the properties and behavior of another class (super-class). It helps to reuse, customize and enhance the existing code. So, it helps to write a code accurately and reduce the development time.
* Polymorphism: Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So, polymorphism means "many forms". A subclass can define its own unique behavior and still share the same functionalities or behavior of its parent/base class. A subclass can have their own behavior and share some of its behavior from its parent class not the other way around. A parent class cannot have the behavior of its subclass.

1. Contrast one programming language to another programming languages based on object-oriented programming languages popularity

- Java is a statically typed and compiled language, and Python is a dynamically typed and interpreted language. This single difference makes Java faster at runtime and easier to debug, but Python is easier to use and easier to read.

Name:Wilkins Aranas

Course & Year: BEET 3C

Date Submitted: April 4, 2022

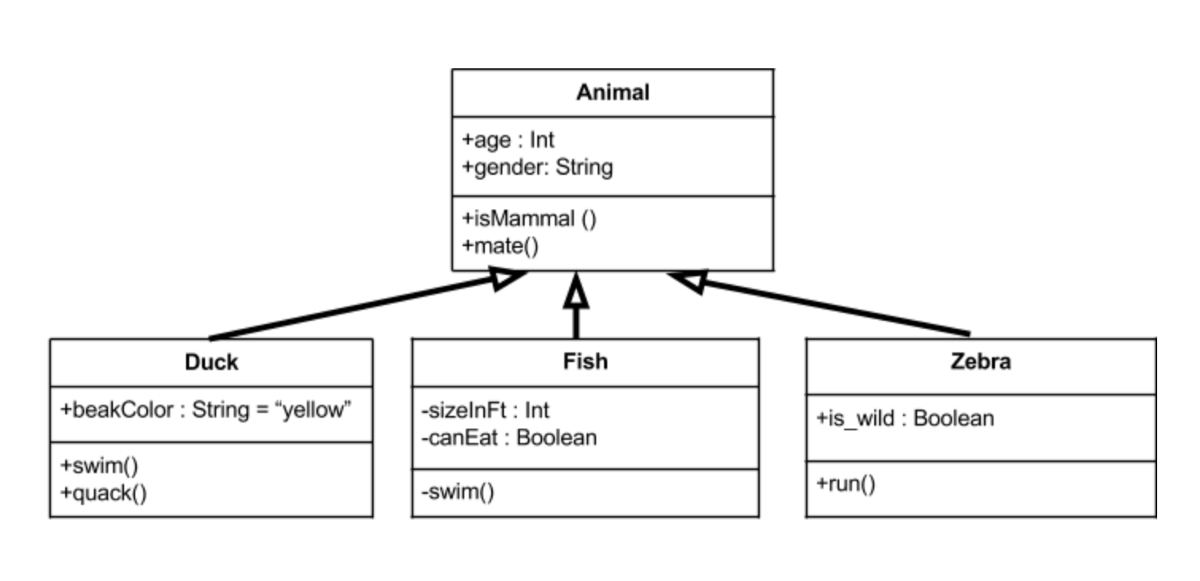
Direction: In one paragraph, answer each of the following questions below. 10 points each question Questions:

1. Distinguish coupling and cohesion in object-oriented programming in terms of diagram or in your own diagram based on its design principles

- The major difference between cohesion and coupling is that cohesion deals with the interconnection between the elements of the same module. But, coupling deals with the interdependence between software modules. Cohesion is defined as the degree of relationship between elements of the same module.

2. Explain responsibility driven design with data driven design

- Responsibility-driven design is in direct contrast with data-driven design, which promotes defining the behavior of a class along with the data that it holds. Data-driven design is not the same as data-driven programming, which is concerned with using data to determine the control flow, not class design.

3.Diagram at least 3 different UML models with system title.

Name:Wilkins Aranas

Course & Year: BEET 3C

Date Submitted: April 4, 2022

Post-Test

Program Description

Make/create a program using each primitive data type. Make a description of the given program. Encircle the program output. Please follow the sequence.

Program Code:

public class Main {

public static void main(String[] args) {

int myNum = 75; // integer (whole number)

float myFloatNum = 5.99f; // floating point number

char myLetter = 'D'; // character

boolean myBool = true; // boolean

String myText = "Hello"; // String

System.out.println(myNum);

System.out.println(myFloatNum);

System.out.println(myLetter);

System.out.println(myBool);

System.out.println(myText);

}

}

Program output:

5

5.99

D

True

Hello

Name:Wilkins Aranas

Course & Year: BEET 3C

Date Submitted: April 4, 2022

Post-Test

Direction: Give what is asked: 10 points Questions:

1. Explain the internal working of the java try-catch block in exception handling.

- Java try block is used to enclose the code that might throw an exception. It must be used within the method. If an exception occurs at the particular statement in the try block, the rest of the block code will not execute. So, it is recommended not to keep the code in try block that will not throw an exception.

1. Explain exception handling and its hierarchy based on its importance

- In Java “an event that occurs during the execution of a program that disrupts the normal flow of instructions” is called an exception. This is generally an unexpected or unwanted event which can occur either at compile-time or run-time in application code.

Name:Wilkins Aranas

Course & Year: BEET 3C

Date Submitted: April 4, 2022

Post-Test

Give what is asked: 10 points in one paragraph, answer each of the following questions. Write your answer on the space provided.

Direction: Question:

1. Explain AWT container and component classes in graphical user interface programming

Direction: Discuss further the graphical user interface program and trace the output. 10 point each.

1. import javax.swing.\*;

class gui{

public static void main(String args[]){

JFrame frame = new JFrame("My First GUI");

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

frame.setSize(300,300);

JButton button = new JButton("Press");

frame.getContentPane().add(button); // Adds Button to content pane of frame

frame.setVisible(true);

}}

Answer =

2. import javax.swing.\*;

class gui {

public static void main(String args[]){

JFrame frame = new JFrame("My First GUI");

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

frame.setSize(300,300);

JButton button1 = new JButton("Press");

frame.getContentPane().add(button1);

frame.setVisible(true);

}}

Midterm requirements

Problem/code 1: Java boolean data type public

class Main {

static void main(String[] args) {

boolean flag = true;

System.out.println(flag); // prints true }

}

|  |
| --- |
| 1.  public class Main {  public static void main(String[] args) {  int myNum = 75; // integer (whole number)  System.out.println(myNum);  Output : 75 // integer data type  }  }  2.  public class Main {  public static void main(String[] args) {  float myFloatNum = 5.99; // floating point number  System.out.println(myFloatNum);  }  }  Output: 5.99 // Float data type  3.  public class Main {  public static void main(String[] args) {  char myLetter = 'D'; // character  System.out.println(myLetter);  }  }  Output : D // character data type |

2: Java byte data type

class Main {

public static void main(String[] args) {

byte range;

range = 124;

System.out.println(range); // prints 124 } }

|  |
| --- |
| 1.  class Main {  public static void main(String[] args) {  byte range;  range = 4;  System.out.println(range); // prints 4  }  }  Output : 4  2.  class Main {  public static void main(String[] args) {  byte range;  range = 8;  System.out.println(range); // prints 8  }  }  Ouput : 8 |

3: Java short data type

class Main {

public static void main(String[] args) {

short temperature;

temperature = -200;

System.out.println(temperature); // prints -200

}

}

Ouput : -200

|  |
| --- |
| 1.  class Main {  public static void main(String[] args) {  short temperature;  temperature = -100;  System.out.println(temperature); // prints -100  }  }  Ouput : -100  2.  class Main {  public static void main(String[] args) {  short temperature;  temperature = -5.0;  System.out.println(temperature); // prints -50  }  }  Ouput : -50 |

4: Java int data type

class Main {

public static void main(String[] args) {

int range = -4250000;

System.out.println(range); // print -4250000

}

}

Output: -4250000

|  |
| --- |
| 1.  class Main {  public static void main(String[] args) {    int range = 21;  System.out.println(range); // print 21  }  }  Output: 21  2.  class Main {  public static void main(String[] args) {    int range = 453;  System.out.println(range); // print 453  }  }  Output: 453 |

5: Java long data type

|  |
| --- |
| class LongExample {  public static void main(String[] args) {    long range = -42332200000L;  System.out.println(range); // prints -42332200000  }  }  Output: -42332200000  2.  class LongExample {  public static void main(String[] args) {    long range = -1233220000011212L; // print -1233220000011212L    System.out.println(range);    }  }  Output: -1233220000011212L; |

6: Java double data type

class Main {

public static void main(String[] args) {

double number = -42.3;

System.out.println(number); // prints -42.3

}

}

Output: -42.3

|  |
| --- |
| 1.  public class Main {  public static void main(String[] args) {  double myNum = 19.99d;  System.out.println(myNum); // prints 19.99  }  }  Output: 19.99  2.  public class Main {  public static void main(String[] args) {  double myPrice = 12.24d;  System.out.println(myPrice); // Print 12.24  }  }  Output: 12.24 |

7: Java float data type

class Main {

public static void main(String[] args) {

float number = -42.3f;

System.out.println(number); // prints -42.3

}

}

|  |
| --- |
| class Main {  public static void main(String[] args) {    float number1 = -42.3f;  float number2 = -47.3f;  System.out.println(number1); // prints -42.3  System.out.println(number2); // prints -47.3  }  }  Ouput:  -42.3  -47.3 |

8: Java char data type

class Main {

public static void main(String[] args) {

char letter = '\u0051';

System.out.println(letter); // prints Q

}

}

Output : Q

class Main {

public static void main(String[] args) {

char letter1 = '9';

System.out.println(letter1); // prints 9

char letter2 = 65;

System.out.println(letter2); // prints A

}}

Output : 9 A

|  |
| --- |
| class Main {  public static void main(String[] args) {    char letter1 = '12';  System.out.println(letter1); // prints 12    char letter2 = 66;  System.out.println(letter2); // prints B  }  }  Output : 12 B |

9. Java Variable Types

Problem/code 9

class A

{

Int x=12;//instance variable

Static int y=13; //static variable

public static void main(String...s)

{

Int z=30; //local variable

System.out.println(x);

System.out.println(y);

System.out.println(z);

}

}

Instruction: Write your comment in paragraph form in the box

|  |
| --- |
| It is an error: non-static variable x cannot be referenced from a static context |

10 Declaring Constant as Private

import java.util.Scanner;

public class ConstantExample1

{

//declaring constant

private static final double PRICE=234.90;

public static void main(String[] args)

{

int unit;

double total\_bill;

System.out.print("Enter the number of units you have used: ");

Scanner sc=new Scanner(System.in);

unit=sc.nextInt();

total\_bill=PRICE\*unit;

System.out.println("The total amount you have to deposit is: "+total\_bill);

}

}

Output :

Enter the number of units you have used: 12

The total amount you have to deposit is: 2818.8

|  |
| --- |
| import java.util.Scanner;  public class ConstantExample1  {  import java.util.Scanner;  public class ConstantExample1  {  //declaring constant  private static final double PRICE=100.90;  public static void main(String[] args)  {  int unit;  double total\_bill;  System.out.print("Enter the number of units you have used: ");  Scanner sc=new Scanner(System.in);  unit=sc.nextInt();  total\_bill=PRICE\*unit;  System.out.println("The total amount you have to deposit is: "+total\_bill);  }  }  Enter the number of units you have used: 15  The total amount you have to deposit is: 1513.5 |

11. ConstantExample2.java

public class ConstantExample2

{

private static final double PRICE=2999;

public static void main(String[] args)

{

System.out.println("Old Price of Iron: "+PRICE);

ConstantExample obj = new ConstantExample();

obj.showPrice();

}

}

class ConstantExample

{

private static final double PRICE=3599;

void showPrice()

{

System.out.print("New Price of Iron: "+PRICE);

}

}

Instruction: Trace the code and write your output in the box

|  |
| --- |
| public class ConstantExample2  {  private static final double PRICE=3999;  public static void main(String[] args)  {  System.out.println("Old Price of Iron: "+PRICE);  ConstantExample obj = new ConstantExample();  obj.showPrice();  }  }  class ConstantExample  {  private static final double PRICE=3599;  void showPrice()  {  System.out.print("New Price of Iron: "+PRICE);  }  }  Output : Old Price of Iron: 3999.0New Price of Iron: 3599.0 |

12. ConstantExample3.java

public class ConstantExample3

{

//declaring PI as constant

public static final double PI= 3.14;

public static void main(String[] args)

{

printValue();

//trying to assign 3.15 in the constant PI

PI = 3.15;

printValue();

}

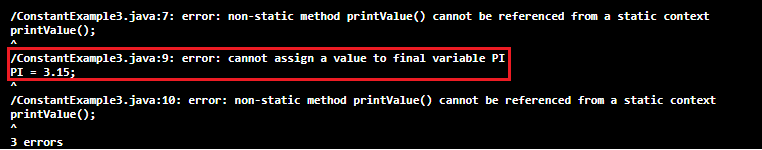
void printValue()

{

System.out.print("The value of PI cannot be changed to " + PI);

}

}



Instruction: Discuss the given output in the box

|  |
| --- |
| It is difficult to allocate a benefit of something steady which the mistakes happens on the grounds that the of its constant state. |