# Section A:

**TK 1143 Program Design 2020/2021 Polymorphism**

1. Based on the following code, can you identify either it is constructor overloaded or method overloaded? Justify your answer.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | public class Kira  {  public Kira(){  //-----> (1)  }  Public void Kira() {  //-----> (2)  }  } |

Answer:

Not both, because only 1 constructor and 1 method

1. a) Understand the following code and predict the output

*Dynamic Polymorphism*

|  |  |  |
| --- | --- | --- |
|  | 1 | **//Class SeaCreature**  public class SeaCreature { public void method1() {  System.out.println("creature 1");  }  public void method2() { System.out.println("creature 2");  }  public String toString() { return "ocean dwelling";  }  }  **// Class Mammal**  public class Mammal extends SeaCreature { public void method1() {  System.out.println("warm blooded");  }  }  **// Class Whale**  public class Whale extends Mammal { public void method1() { |
| 2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | |

|  |  |
| --- | --- |
| 22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | System.out.println("spout");  }  public String toString() { return "BIG!";  }  }  **// Class Squid**  public class Squid extends SeaCreature { public void method2() {  System.out.println("tentacles");  }  public String toString() { return "squid";  }  } |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **// Class SeaCreatureApp**  public class SeaCreatureApp {  public static void main (String []args){  SeaCreature[] elements = {new Squid(), new Whale(), new SeaCreature(),new Mammal()};  for (int i = 0; i < elements.length; i++) { System.out.println(elements[i]); elements[i].method1(); elements[i].method2(); System.out.println();  }  }  } |

Answer:

squid

creature 1

tentacles

BIG!

spout

creature 2

ocean dwelling

creature 1

creature 2

ocean dwelling

warm blooded

creature 2

* 1. Based on your observations, does the program uses an overloading OR overriding method? Explain your answer.

Overriding method. Because every subclass have same method. There is a method that is used multiple times in different subclasses.

* 1. Draw a UML class diagram for java code program in question 2(a).

Letter

Description automatically generated

1. Identify **NINE** syntax errors in the following Java program segment and modify the program to remove the errors.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39 | **//Class Student**  public class Student { protected String name;  protected String matricNum;  public Student(String n, String m) { name = n;  matricNum = m;  }  public void displayDetails() { System.out.println("Name: " + name); System.out.println("Matric #: " + matricNum);  }  }  **//Class UnderGradStudent**  public class UnderGradStudent extends Student {  private String program;  public UnderGradStudent(String n, String m, String k) {  super (n,m);  program = k;  }  }  **// class PostGradStudent**  public class PostGradStudent extends Student { private String supervisor;  public PostGradStudent(String n, String m, String s)  {  super(n,m);  supervisor=s;  }  public void displayDetails() { System.out.println("Name: "+name); System.out.println("Matric #: "+matricNum); System.out.println("Supervisor: "+supervisor);  }  } |

Answer :

*\*\* make a correction on code*

1. Please complete code based on the output given.

**class** Bird {

//a) the constructor class Bird

}

**public class** BirdApp {

**public static void** main(String[] args) {

//b instantiate the Birds Object

myBird.fly(); myBird.fly(10000); myBird.fly("eagle", 10000);

}

}

Output :

The bird is flying.

The bird is flying 10000 feet high. The eagle is flying 10000 feet high.

Answer a):

public Bird() {}

Answer b):

myBird = new Bird ()

public void fly() {

System.out.println(“The bird is flying.”);

}

public void fly ( int feet){

System.out.println(“The bird is flying” +feet+ “feet high.”);

}

public void fly ( String name, int feet){

System.out.println(“The” +name+ “is flying” +feet+ “feet high.”);

**}**

1. Why is following code showing compile error?

**public class** AnimalApp{

**public static void** main(String[] args){

Animal[] a = {**new** Animal("Animal"),**new** Dog("Dog"),**new** Cat("Cat"),

**new** Lion("Lion")};

**for**(**int** x=0 ; x<a.length; x++) { a[x].eat();}

}

}

**class** Animal{

**protected** String animal\_type;

**public** Animal(String type) { animal\_type=type;

}

**void** eat(){

System.***out***.println(animal\_type + " eating...");}

}

**class** Dog **extends** Animal{

**void** eat(){System.***out***.println(animal\_type + " eating bone...");}

}

**class** Cat **extends** Animal{

**void** eat(){System.***out***.println(animal\_type + " eating fish...");}

}

**class** Lion **extends** Animal{

**void** eat(){System.***out***.println(animal\_type + " eating meat...");}

}

Answer a):

All subclass have no super constructor

b) Fix the error from the following code and display the output. Answer b):

**public** **class** AnimalApp {

**public** **static** **void** main(String[] args){

Animal[] a = {**new** Animal("Animal"),**new** Dog("Dog"),**new** Cat("Cat"),

**new** Lion("Lion")};

**for**(**int** x=0 ; x<a.length; x++) { a[x].eat();}

}}

**class** Animal{

**protected** String animal\_type;

**public** Animal(String type) {

animal\_type=type;

}

**void** eat(){

System.***out***.println(animal\_type + " eating...");}

}

**class** Dog **extends** Animal{

**public** Dog (String type) {

**super**(type); }

**public** **void** eat(){

System.***out***.println(animal\_type + " eating bone...");}

}

**class** Cat **extends** Animal{

**public** Cat (String type) {

**super**(type); }

**public** **void** eat(){

System.***out***.println(animal\_type + " eating fish...");}

}

**class** Lion **extends** Animal{

**public** Lion (String type) {

**super**(type); }

**public** **void** eat(){

System.***out***.println(animal\_type + " eating meat...");}

}

Answer b (output):

Animal eating...

Dog eating bone...

Cat eating fish...

Lion eating meat...

1. a) Based on the following code, can you find any problems? Explain your answer.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | **class** Loan {  **public double** getRateOfInterest() {  **return** 4.5;}  }  **class** Car **extends** Loan {  **public double** getRateOfInterest() {  **return** 3.6;}  **public** String toString() {  **return** "Car";}  }  **class** House **extends** Loan {  //c  }  **class** Land **extends** Loan {  **public double** getRateOfInterest() {  **return** 6.5;}  **public** String toString() {  **return** "Land";}  }  **public class** LoanApp {  **public static void** main(String args[]) {  Car[] loan= {**new** Car(), **new** House(), **new** Land()};  **for** (**int** i=0;i<loan.length;i++) { System.***out***.println("Interest Rate for " +loan[i] + " is " +loan[i].getRateOfInterest() + "%");  }  }  } |

Answer a):

Yes, there is a problem. In line 30 the constructor should name for super class not subclass.

1. What is the output for the code after you fix the problem in a. Explain why it happen. Answer b):

Interest Rate for Car is 3.6%

Interest Rate for House is 4.5%

Interest Rate for Land is 6.5%

1. Complete the class House so that the correct output can be generated.

Answer c):

public String toString() {

return "House";

}

1. In your opinion, what is the purpose of toString Method in this code. Are we overload the method or override it? Share your idea.

Answer d):

The purpose of toString Method in this code is to return the value for particular class in String format.The method is override because the toString method was use multiple time in different subclass