**Homework 2**

1. (10 pts) What are the diagrams defined in the UML Standard. Give a one or two sentence description of each one.

**- Class diagram: Class diagram consists of classes, interfaces, associations, and collaboration and they represent the object-oriented view of a system, which is static in nature.  
- Object diagram: Object diagrams are a set of objects and their relationship is close to class diagrams which represent the static view of the system.  
- Component diagram: Component diagrams represent a set of components and their relationships. These components consist of classes, interfaces, or collaborations.  
- Deployment diagram: Deployment diagrams are a set of nodes and their relationships. These nodes are physical entities where the components are deployed. They are used for visualizing the deployment view of a system.  
  
2) Behavioral Diagrams: Behavioral diagrams capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing or moving parts of a system. There are five types of behavioral diagrams: use case diagram, sequence diagram, collaboration diagram, statechart diagram, and activity diagram.  
  
- Use case diagram: A use case represents a particular functionality of a system. Hence, use case diagram is used to describe the relationships among the functionalities and their internal/external controllers.  
- Sequence diagram: A sequence diagram is an interaction diagram. The diagram deals with sequences, which are the sequence of messages flowing from one object to another.  
- Collaboration diagram: Collaboration diagram represents the structural organization of a system and the messages sent or received. The purpose of collaboration diagram is to visualize the organization of objects and their interaction.  
- Statechart diagram: Statechart diagram is used to represent the event driven state change of a system. It is used to visualize the reaction of a system by internal/external factors.  
- Activity diagram: Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.**  
  
2. (10 pts) Given the following code, how should the toString methods in the classes H2ClassA and H2ClassB be written to give the indicated output and take advantage of the natural toString method in H2ClassB?

1  import java.util.ArrayList;  
2   
3  public class H2ClassA {  
4    ArrayList <H2ClassB> list = new ArrayList <H2ClassB> ();  
5   
6    public static void main (String args []) {  
7      H2ClassA y = new H2ClassA ();  
8      int [] v = {4, 3, 7, 5, 99, 3};  
9      for (int m: v)   
10       y.list.add (new H2ClassB (m));  
11     System.out.println (y);  
12   } // end main  
13   
14 } // end class H2ClassA  
15   
16 class H2ClassB {  
17   int x;  
18   H2ClassB (int a) { x = a;}  
19 } // end H2ClassB

OUTPUT:

4 3 7 5 99 3

**Answer:**   
**Overridden toString() methods should be included in both class H2ClassA and H2ClassB.   
  
1) Include the following toString method within class H2ClassB  
  
 @Override  
  
 public String toString()   
 {  
 return Integer.toString(x);  
 }  
   
2) Include the following toString method within class H2ClassA   
  
 @Override  
  
 public String toString()  
 {  
 String str = "";  
  
 for(H2ClassB t : list)  
 {  
 str = str + t.toString() + " ";  
 }  
 return str;  
 }**

3. (10 pts) How can the following code be corrected? Give at least two good answers.

1 public class H2ClassC {  
2   H2ClassC (int a) {}  
3 } // end class H2ClassC  
4   
5 class H2ClassD extends H2ClassC{  
6 } // end class H2ClassD

**Answer:**

**1) Since there is no default constructor in class H2ClassC and class H2ClassC’s pre-defined constructor takes parameters, the child class H2ClassC should define an explicit constructor by calling super() with a corresponding argument.   
  
public class H2ClassC   
{  
 H2ClassC (int a) {}  
  
} // end class H2ClassC  
  
class H2ClassD extends H2ClassC  
{   
 H2ClassD(int a)   
 {  
 super(a); //Explicit constructor call added  
 }  
  
} // end class H2ClassD   
  
2) If class H2ClassC has a default constructor, there will be an implicit call to super() with no argument for all child classes of H2ClassC. The child class H2ClassD can still call super() explicitly.  
  
public class H2ClassC**

**{  
 H2ClassC() {} //Default constructor added**

**H2ClassC (int a) {}  
  
} // end class H2ClassC  
  
 class H2ClassD extends H2ClassC{  
  
 H2ClassD() {  
 super(); //Default constructor added  
 }  
 } // end class H2ClassD**

4. (10 pts) Why does the following code give a compiler error? How should it be fixed?

1  public class H2ClassE {  
2    int x, y, z;  
3   
4    H2ClassE (int a) {  
5      x = a;  
6      this (5, 12);  
7    }  
8   
9    H2ClassE (int b, int c) {  
10     y = b;  
11     z = c;  
12   }  
13 } // end class H2ClassE

**Answer:  
Line 6 (this(5,12);) should be placed before line 5 like below, because constructor call needs to be the first statement in constructor.  
  
 H2ClassE(int a)   
 {  
 this(5, 12); //Constructor call should be made first  
 x = a;  
 }**

5. (10 pts) What is wrong with the following declaration? How should it be fixed?

public static final int myNumber = 17.36;  
  
**Answer:**

**The variable myNumber is declared as int, but a double value with decimal point is assigned. Therefore, type mismatch occurs. It can be fixed by either declaring the variable as double, or assign an integer value to the variable without changing its type.  
  
1) public static final double mydouble = 17.36;  
2) public static final int myNumber = 17;**

6. (10 pts) What is wrong with the following code? How should it be fixed?

1 public class H2ClassG {  
2   final int x;  
3   
4   H2ClassG () {}  
5   H2ClassG (int a) {x = a;}  
6 } // end class H2ClassG

**Answer:**  
**Once the code is compiled, it generates an error message saying “variable x might not have been initialized”. Assigning a value to the final variable x can fix the problem.  
  
public class H2ClassG {  
  
 final int x;   
  
 H2ClassG() {  
 x = 1;  
 }   
 H2ClassG(int a) {  
 x = a;  
 }   
} // end class H2ClassG**

7. (10 pts) What is wrong with the following code? How should it be fixed?

1 public class H2ClassH {  
2   final int x;  
3   
4   int H2ClassH () {  
5     if (x == 7) return 1;  
6     return 2;  
7   } // end  
8 } // end class H2ClassH

**Answer:**  
**The code above generates the following error message: “variable x not being initialized in the default constructor”. The final variable x should be assigned first either when it is declared or within the constructor.  
  
public class H2ClassH {  
 final int x = 7;   
  
 int H2ClassH() {  
  
 if (x == 7)  
  
 return 1;  
  
 return 2;  
  
 } // end  
}**

8. (10 pts) What is wrong with the following code? x should be given a value of 24. What are two ways this can be legally accomplished?

1 public class H2ClassI {  
2   final int x;  
3   
4   public static void main (String args []) {  
5     H2ClassI h = new H2ClassI ();  
6     h.x = 24;  
7   } // end main  
8 } // end class H2ClassI

**Answer:**  
**The fixed value of x can be assigned either when it is declared and within the constructor and the value remains constant throughout the class. In order to change the value of x, x can be declared as a class variable without “final” keyword or the value of x can be assigned within the class constructor.  
  
1) Remove “final” keyword.   
  
 public class H2ClassI {  
 int x;  
 public static void main (String args []) {  
 H2ClassI h = new H2ClassI ();  
 h.x = 24;  
 } // end main  
} // end class H2ClassI  
  
2) Keep “final” keyword and create a class constructor that assigns 24 to x.  
  
public class H2ClassI  
 {  
 final int x;  
 public H2ClassI(int n){  
 x = n;  
 }  
 public static void main (String args []){  
 H2ClassI h = new H2ClassI (24);  
 } // end main  
 } // end class H2ClassI**

9. (10 pts) What is wrong with the following code? Give two effective ways to fix it.

1  import javax.swing.\*;  
2  import java.awt.event.\*;  
3   
4  public class H2ClassJ extends JFrame {  
5    public static final long serialVersionUID = 22;  
6   
7    public H2ClassJ () {  
8      addMouseListener (new MouseListener () {  
9        public void mouseClicked (MouseEvent e) {}  
10     });  
11   } // end constructor  
12   
13 } // end class H2ClassJ

**Answer:  
  
Class MouseListener is an abstract class, and five abstract methods such as mousePressed(MouseEvent e) in MouseListener interface must be overridden within the class in order to compile. Or, MouseAdapter class can be used without having to override all methods.  
  
1) Override all methods within Class MouseListener  
 public H2ClassJ() {  
 addMouseListener(new MouseListener() {  
 public void mouseClicked(MouseEvent e) {  
 }  
 //Override the rest of 4 methods in the MouseListener interface  
 public void mousePressed(MouseEvent e) { }  
  
 public void mouseReleased(MouseEvent e) { }  
  
 public void mouseEntered(MouseEvent e) { }  
  
 public void mouseExited(MouseEvent e) { }  
 });  
 } // end constructor  
  
2) Use MouseAdapter   
  
 public H2ClassG () {  
 addMouseListener (new MouseAdapter () {  
 public void mouseClicked (MouseEvent e) {}  
   
 });  
} // end constructor**

10. (10 pts) Why does the following code give a compiler warning? (Use javac -Xlint) How should it be fixed?

1 import javax.swing.\*;  
2   
3 public class H2ClassK {  
4   String [] sa = {"a", "b", "c"};  
5   JComboBox jcbA = new JComboBox (sa);  
6 } // end class H2ClassK

**Answer:**  
**A compiler warning occurs because JComboBox is not parameterized. The error can be fixed by specifying the type of elements within the JcomboBox as String.  
  
import javax.swing.\*;  
  
 public class H2ClassK {  
 String [] sa = {"a", "b", "c"};  
 JComboBox<String> jcbA = new JComboBox<String> (sa);  
  
 } // end class H2ClassK**

**Grading Rubric:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Meets** | **Does not meet** |
| Problem 1 | **10 points** Gives a one or two sentence description of each standard UML diagram. | **0 points** Does not give a one or two sentence description of each standard UML diagram. |
| Problem 2 | **10 points** Explains how the toString methods in the classes H2ClassA and H2ClassB be written to give the indicated output and take advantage of the natural toString method in H2ClassB. | **0 points** Does not explains how the toString methods in the classes H2ClassA and H2ClassB be written to give the indicated output and take advantage of the natural toString method in H2ClassB. |
| Problem 3 | **10 points** Provides at least two good answers explaining how the code can be corrected. | **0 points** Does not provide at least two good answers explaining how the code can be corrected. |
| Problem 4 | **10 points** Explains why the code gives a compiler error.  Explains how the code should be fixed. | **0 points** Does not explain why the code gives a compiler error.  Does not explain how the code should be fixed. |
| Problem 5 | **10 points** Explains what is wrong with the declaration.  Explains how the code should be fixed. | **0 points** Does not explain what is wrong with the declaration.  Does not explain how the code should be fixed. |
| Problem 6 | **10 points** Explains what is wrong with the code.  Explains how the code should be fixed. | **0 points** Does not explain what is wrong with the code.  Does not explain how the code should be fixed. |
| Problem 7 | **10 points** Explains what is wrong with the code.  Explains how the code should be fixed. | **0 points** Does not explain what is wrong with the code.  Does not explain how the code should be fixed. |
| Problem 8 | **10 points** Explains what is wrong with the code.  Explains two ways x could be given a values of 24 legally. | **0 points** Does not explain what is wrong with the code.  Does not explain two ways x could be given a values of 24 legally. |
| Problem 9 | **10 points** Explains what is wrong with the code.  Explains 2 effective ways the code could be fixed. | **0 points** Does not explain what is wrong with the code.  Does not explain 2 effective ways the code could be fixed. |
| Problem 10 | **10 points** Explains why the code gives a compiler warning.  Explains how it should be fixed. | **0 points** Does not explain why the code gives a compiler warning.  Does not explain how it should be fixed. |