CS 1324 Spring 2021 Homework 14 Classes with Objects

Jordan McFadden

TOTAL POINTS

16 / 20

QUESTION 1

1 Question 1 8.5 / 9

- **0 pts** Correct
- **1 pts** Partial
- **2 pts** Partial
- ✓ **- 0.5 pts** **Partial**

Please see the solutions in canvas, some '-' should have been '+'

QUESTION 2

2 Question 2 4 / 6

- **0 pts** Correct
- **0.5 pts** Partial
- **1 pts** Partial
- **1.5 pts** Partial
- ✓ **- 2 pts** **Partial**
- **3 pts** Partial
- **6 pts** Not answered

Please see the solutions in Canvas

QUESTION 3

3 Question 3 3.5 / 5

- **0 pts** Correct
- **0.5 pts** Partial
- **1 pts** Partial
- ✓ **- 1.5 pts** **Partial**
- **2.5 pts** Partial

Please see the solutions in Canvas

Homework 14: Classes with Objects

CS 1323/4 Spring 2021

Name: Jordan McFadden

Student ID (usually 112-XXX-XXXX or 113-XXX-XXXX): 113502650

1. (9 points) Create a UML diagram for part of the Double class by reading the Java API. Your diagram should have two static data, three instance methods, two static methods, two constructors and the private data. The Double class contains one double called value that is private. This is not visible in the API. It is not possible to do underline in Fillable PDF documents, so just put a comment //static, after each static variable.

Double
- value: double - MAX_Value: double //static - MIN_VALUE: double //static

+Double: (value: double)
+Double: (s: String)
+compare(d1: double, d2: double) //static
+max(a: double, b: double) //static
+compareTo(anotherDouble: Double)
+doubleValue()
+intValue()

2. (6 points) The Email class is described in UML below.

Email
-subject: String -sentDate: String -importance: int +HIGH_IMPORTANCE: int +LOW_IMPORTANCE: int +REGULAR_IMPORTANCE: int
+Email(subject: String, sentDate: String, importance: int) +setSubject(subject: String): void +setImportance(importance: int): void +getSubject(): String +getSentDate(): String +getImportance(): int

Draw a memory diagram that shows the code below being executed. Pay attention to the difference between class and instance data. Remember that class data is underlined in UML. Class data is not shown in memory diagrams.

Email hello = new Email("Hello world", "04/28/2021", Email.HIGH_IMPORTANCE);

hello.setSubject("Goodbye");

hello.setImportance(Email.REGULAR_IMPORTANCE);

Email howAreYou = new Email("How are you?", "04/28/2021", Email.LOW_IMPORTANCE);

main stack frame

Identifier	Address	Contents
hello	100	1000
howAreYou	101	1003
	102	
	103	
	104	
	105	

heap

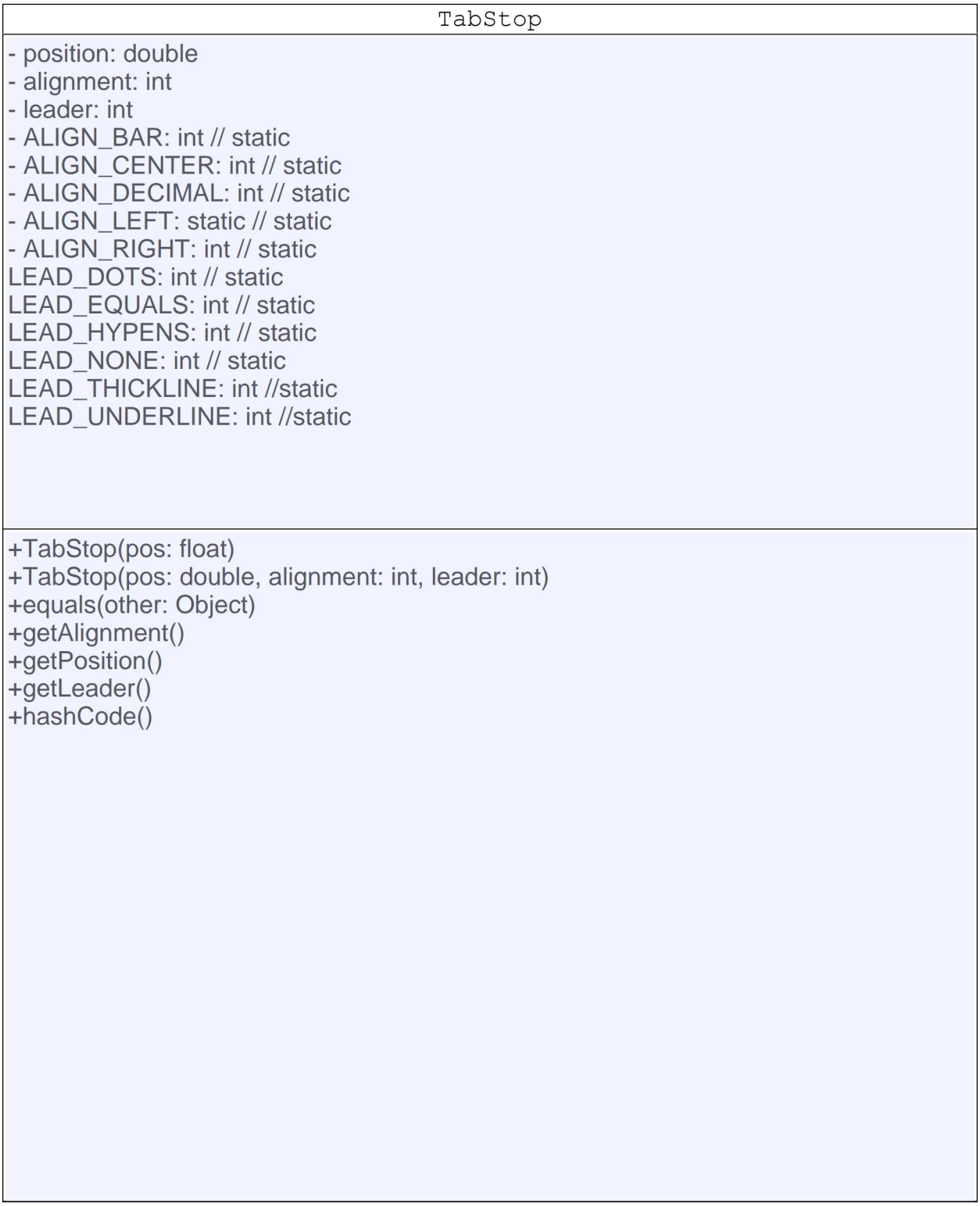
Identifier	Address	Contents
0	1000	Hello World, Goodbye
1	1001	04/28/2021
2	1002	int: high, int: regular
0	1003	How are you?
1	1004	04/28/2021
2	1005	int: low
	1006	
	1007	
	1008	
	1009	
	1010	
	1011	
	1012	
	1013	

3. (5 points) Create a UML diagram for the class expressed in code below in the table on the last page. This class was inspired by a class in the API (although the implementation is, ahem, rather different than the real class.

```
public class TabStop
{
	private double position;
	private int alignment;
	private int leader;

	public static final int ALIGN_BAR = 1;
	public static final int ALIGN_CENTER = 2;
	public static final int ALIGN_DECIMAL = 3;
	public static final int ALIGN_LEFT = 4;
	public static final int ALIGN_RIGHT = 5;

	public static final int LEAD_DOTS = 6;
	public static final int LEAD_EQUALS = 7;
	public static final int LEAD_HYPENS = 8;
	public static final int LEAD_NONE = 9;
	public static final int LEAD_THICKLINE = 10;
	public static final int LEAD_UNDERLINE = 11;

	public TabStop(float pos)
	{
		position = pos;
	}

	public TabStop(double pos, int align, int leader)
	{
		this.position = pos;
		this.alignment = align;
		this.leader = leader;
	}

	public boolean equals(Object other)
	{
		// TabStop objects can't be compared to other classes
		if (! (other instanceof TabStop))
			return false;

		TabStop obj = (TabStop) other;
		if (obj.alignment != this.alignment)
			return false;
		if (obj.position != this.position)
			return false;
```

```
            if (obj.leader != this.leader)
                  return false;

            // If we get here all fields match
            return true;
      }

      public int getAlignment()
      {
            return alignment;
      }

      public int getLeader()
      {
            return leader;
      }

      public double getPosition()
      {
            return position;
      }

      public int hashCode()
      {
            return super.hashCode();
      }
}
```

The diagram is on next page—it is not possible to underline in Fillable PDF documents, so put //static in a comment after each static field.


TabStop
- position: double
- alignment: int
- leader: int
- ALIGN_BAR: int // static
- ALIGN_CENTER: int // static
- ALIGN_DECIMAL: int // static
- ALIGN_LEFT: static // static
- ALIGN_RIGHT: int // static
LEAD_DOTS: int // static
LEAD_EQUALS: int // static
LEAD_HYPENS: int // static
LEAD_NONE: int // static
LEAD_THICKLINE: int //static
LEAD_UNDERLINE: int //static
+TabStop(pos: float)
+TabStop(pos: double, alignment: int, leader: int)
+equals(other: Object)
+getAlignment()
+getPosition()
+getLeader()
+hashCode()