CS 1324 Spring 2021 Homework 10 Methods and References

Jordan McFadden

TOTAL POINTS

17 / 20

QUESTION 1

1 Question 1 **5 / 5**

- ✓ **- 0 pts Correct**
- **- 1 pts** Incorrect main stack
- **- 1 pts** Incorrect heap
- **- 1 pts** Incorrect setAlternateValues stack
- **- 1 pts** Incorrect output based on values
- **- 5 pts** Missing

It is good practice in this class to include all changes in the tracing

QUESTION 2

2 Question 2 **4 / 5**

- **- 0 pts** Correct
- **- 1 pts** Incorrect main stack
- **- 1 pts** Incorrect heap
- ✓ **- 1 pts Incorrect setAlternateValues stack**
- **- 1 pts** Incorrect output based on values
- **- 5 pts** Missing
- **- 0 pts** See comment

1 missing index

QUESTION 3

3 Question 3 **4 / 5**

- **- 0 pts** Correct
- **- 1 pts** Incorrect main stack
- ✓ **- 1 pts Incorrect setAlternateValues stack**
- **- 1 pts** Incorrect heap
- **- 1 pts** Incorrect output based on values
- **- 5 pts** Missing
- **- 0 pts** See comment

2 missing index

QUESTION 4

4 Question 4 **4 / 5**

- **- 0 pts** Correct
- **- 1 pts** Incorrect main stack
- ✓ **- 1 pts Incorrect setAlternateValues stack**
- **- 1 pts** Incorrect heap
- **- 1 pts** Incorrect output based on values
- **- 5 pts** Missing
- **- 0 pts** See comment

3 missing index

Homework 10: Methods and References

CS 1323/4 Spring 2021

Name: Jordan McFadden

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

1. (5 points) Trace the execution of the code below in the memory diagram. **Based on your diagram, give the value that will be printed out when the program is executed.**

```
public class TraceMe1
{

      public static void main(String[] args)
      {
            int[] array = {1, 3, 5, 2, 4};
            setAlternateValues(array, 0);
            System.out.println(Arrays.toString(array));
      }

      public static void setAlternateValues(int[] data, int target)
      {
            for (int index=0; index < data.length; index += 2)
                        // The increment means index = index + 2
            {
                  data[index] = target;
            }
      }
}
```

Output: {0, 3, 0, 2, 0}

main stack frame		
Identifier	**Address**	**Contents**
array	100	1000
	101	
	102	
	103	
	104	

setAlternateValues stack frame		
Identifier	**Address**	**Contents**
data	200	1000
target	201	0
	202	
	203	
	204	

Heap		
Identifier	**Address**	**Contents**
0	1000	0
1	1001	3
2	1002	0
3	1003	2
4	1004	0
array.length	1005	5
	1006	
	1007	
	1008	
	1009	
	1010	
	1011	
	1012	

2. (5 points) Trace the execution of the code below in the memory diagram. **Based on your diagram, give the value that will be printed out when the program is executed.**

```java
public class TraceMe2
{

        public static void main(String[] args)
        {
                int[] array = {1, 3, 5, 2, 4};
                setAlternateValues(array, 0);
                System.out.println(Arrays.toString(array));
        }

        public static void setAlternateValues(int[] data, int target)
        {
                int[] result = new int[data.length];

                for (int index=0; index < data.length; ++index)
                {
                        if (index % 2 == 0)
                                result[index] = target;
                        else
                                result[index] = data[index];
                }

                data = result;
        }
}
```

Output: {1,3,5,2,4}

main stack frame		
Identifier	**Address**	**Contents**
array	100	1000
	101	
	102	
	103	
	104	

setAlternateValues stack frame		
Identifier	**Address**	**Contents**
data	200	1006
target	201	0
result	202	1006
	203	
	204	
	205	
	206	

Heap		
Identifier	**Address**	**Contents**
0	1000	1
1	1001	3
2	1002	5
3	1003	2
4	1004	4
array.length	1005	5
0	1006	0
1	1007	3
2	1008	0
3	1009	2
4	1010	0
result.length	1011	5
	1012	

3. (5 points) Trace the execution of the code below in the memory diagram. **Based on your diagram, give the value that will be printed out when the program is executed.**

```
public class TraceMe3
{

	public static void main(String[] args)
	{
		int[] array = {1, 3, 5, 2, 4};
		setAlternateValues(array, 0);
		System.out.println(Arrays.toString(array));
	}

	public static int[] setAlternateValues(int[] data, int target)
	{
		int[] result = new int[data.length];

		for (int index=0; index < data.length; ++index)
		{
			if (index % 2 == 0)
				result[index] = target;
			else
				result[index] = data[index];
		}

		return result;
	}
}
```

Output: {1,3,5,2,4}

main stack frame		
Identifier	**Address**	**Contents**
array	100	1000
	101	
	102	
	103	
	104	

setAlternateValues stack frame		
Identifier	**Address**	**Contents**
data	200	1000
target	201	0
result	202	1006
	203	
	204	

Heap		
Identifier	**Address**	**Contents**
0	1000	1
1	1001	3
2	1002	5
3	1003	2
4	1004	4
array.length	1005	5
0	1006	0
1	1007	3
2	1008	0
3	1009	2
4	1010	0
result.length	1011	5
	1012	

4. (5 points) Trace the execution of the code below in the memory diagram. **Based on your diagram, give the value that will be printed out when the program is executed.**

```
public class TraceMe4
{

        public static void main(String[] args)
        {
                int[] array = {1, 3, 5, 2, 4};
                array = setAlternateValues(array, 0);
                System.out.println(Arrays.toString(array));
        }

        public static int[] setAlternateValues(int[] data, int target)
        {
                int[] result = new int[data.length];

                for (int index=0; index < data.length; ++index)
                {
                        if (index % 2 == 0)
                                result[index] = target;
                        else
                                result[index] = data[index];
                }

                return result;
        }
}
```

Output: {0,3,0,2,0}

main stack frame		
Identifier	**Address**	**Contents**
array	100	1006
	101	
	102	
	103	
	104	

setAlternateValues stack frame		
Identifier	**Address**	**Contents**
data	200	1000
target	201	0
result	202	1006
	203	
	204	

Heap		
Identifier	**Address**	**Contents**
0	1000	1
1	1001	3
2	1002	5
3	1003	2
4	1004	4
array.length	1005	5
0	1006	0
1	1007	3
2	1008	0
3	1009	2
4	1010	0
result.length	1011	5
	1012	