

Name _____ TA _____

COMP114, Test #2.

Tuesday, April 9, 2002

Pledge: I have neither given nor received unauthorized aid on this exam.

(signed) _____

Please write your answers on the exam form. Closed book and notes.

There are 9 pages on this exam. The last page is scratch paper for your use. Feel free to tear the last page off.

Point values are in parentheses.

Write your name and that of your TA on each page (just in case the pages come apart).

You have the whole class period (75 minutes) to complete this exam.

Points will not be counted off for small syntax errors.

There are no syntax errors in the program fragments printed below.

Part I (54 pts. total)

1. (4) Can I create an instance of an abstract class (yes or no)?

NO

2. (3) What is the greatest advantage of a Vector over an array?

It can grow.

3. Consider the following method in a console application

```
public static void example(int x){
    int array[] = {0, 1, 2};
    try{
        System.out.println(array[x]);
    }
    catch(Exception e){
        System.out.println("Caught an exception!");
    }
    finally{
        System.out.println("Finally");
    }
}
```

a. (4) Please write down what will be printed if this method is called as `example(1);`

1
Finally

b. (4) What will be printed if this method is called as `example(3);`

Caught an exception!
Finally

4. (4) What is meant by the *base case* for recursion (choose one)? **b**

- a. A switch statement to determine which path to follow.
- b. A case that does no further recursion.
- c. A case that determines how to divide the recursion evenly.
- d. The case that begins the recursion.

Name _____ TA _____

5. (15) Write a *recursive* method to sum the numbers from 1 to n. I've given you the method name and parameter.

```
public int sumNumbers(int n){  
  
    if(n < 1)  
        return 0;  
    else  
        return(sumNumbers(n - 1) + n);  
  
}
```

6. (4) Which of these sorting algorithm(s) would I use if my array was already mostly sorted? **a**
- a. Insertion sort
 - b. Merge sort
 - c. Quicksort
7. (4) Please label each as a comparison sort or not.
- a. Bubble sort **Comparison**
 - b. Merge sort **Comparison**
 - c. Radix sort **Not Comparison**
 - d. Quicksort **Comparison**

Name _____ TA _____

8. (4 pts. each, total 8) Write the propositional calculus expression for

a. Not both x and y are greater than z.

$(x \leq z) \parallel (y \leq z)$ **also: $\neg((x > z) \ \&\& \ (y > z))$**

b. The entries of B[0..10] are sorted in increasing order.

($\forall i: 0 \leq i < 100: B[i] \leq B[i+1]$)

also ($\forall i, j: 0 \leq i < j \leq 100: B[i] \leq B[j]$) also ($\forall i, j: 0 \leq i, j \leq 100: (i < j) \Rightarrow (B[i] < B[j])$)

9. (2 pts. each blank, total 4) Fill in the blanks (with the appropriate letter) on this description of the delegation model used by most modern graphical user interfaces.

A user action triggers an **D** , which causes the **B** method of the user code to be called

- | | |
|---------------|-------------|
| a. interfaces | e. main |
| b. handler | f. register |
| c. abstract | g. supply |
| d. event | h. alert |

Part II (21 points)

Here is a very simple sorting method. The code is used for questions 10, 11, and 12.

```
public static void simpleSort(int[] array){
    int i, j;
    for (i = 0; i < array.length - 1; i++){
        for (j = i + 1; j < array.length; j++){
            if (array[j] < array[i])
                swap(array, i, j);
            for(int k = 0; k < array.length; k++)
                System.out.print(array[k] + " ");
            System.out.println("i = " + i + " j = " + j);
        }
    }
}
```

I call this method with an array containing the following integers

9, 5, 3, 11

as follows

```
public static void main(String[] args){
    int a[] = {9, 5, 3, 11};
    simpleSort(a);
}
```

10. (12) Show what is printed on the console.

```
5 9 3 11 i = 0 j = 1
3 9 5 11 i = 0 j = 2
3 9 5 11 i = 0 j = 3
3 5 9 11 i = 1 j = 2
3 5 9 11 i = 1 j = 3
3 5 9 11 i = 2 j = 3
```

Name _____ TA _____

11. (4) What is the loop invariant for the *outer* loop? You can write the invariant in English or in formal mathematical notation, or both.

In English: all array elements from 0 to i are sorted and are less than elements from i to the end of the array.

$(\forall m \forall n: 0 \leq m < n < i: \text{array}[m] \leq \text{array}[n])$

and

$(\forall p \forall q: 0 \leq p < i \leq q < \text{array.length}: \text{array}[p] \leq \text{array}[q])$

12. (5) What is the worst-case time complexity of the algorithm (ignoring the printing, which is just for debugging purposes)?

$O(n^2)$

Name _____ TA _____

Part III Searching (20 points)

13. (4) What is the time complexity of linear search? **$O(n)$**

14. (4) What is the time complexity of binary search? **$O(\log n)$**

15. (12) Write a search to find a number in an unsorted array containing positive integers. Return the index in the array that contains the target, or -1 if it cannot be found. I've supplied the method declaration.

```
public static int search(int target, int array[]){
```

```
    for(int i = 0; i < array.length; i++)
        if(array[i] == target)
            return i ;
    return -1;
```

```
}
```

Name _____ TA _____

Part IV (1 point each, 5 points total)

Match the sorting algorithms with their performance characteristics.

16. Insertion sort **c**

17. Selection sort **a**

18. Quicksort **b**

19. Merge sort **d**

20. Radix sort **e**

a. $O(n^2)$ worst and best case.

b. $O(n^2)$ worst case, $O(n \log n)$ best case.

c. $O(n^2)$ worst case, $O(n)$ best case.

d. $O(n \log n)$ worst and best case.

e. $O(n)$ worst and best case.

f. None of the above.

SCRATCH PAPER