

# ECE122 Introduction to ECE II

Spring 2007

2nd Midterm Examination

(120 minutes, closed book)

NAME: \_\_\_\_\_

Student ID: \_\_\_\_\_

Note that any questions on writing code must be answered using Java topics covered in lectures 1- 15

Question	Score
1 (20 points )	
2 (15 points)	
3 (22 points )	
4 (15 points )	
5 (13 points)	
6 (15 points)	
<b>Total</b>	

1. ( 20 points )

Create the following methods:

a) (10 points) Write a method **SwapArrays** that takes two 2D integer arrays as parameters and swaps their contents *by swapping their row references*. The method **should not** copy the data of one array into the other. Instead, it should *swap row references*. You can assume that the arrays have the same number of rows but you may not assume that they have the same number of columns. The method doesn't return anything.

```
Public void SwapArrays(int[][] a, int[][] b)
{
int[] c;
for(int i=0;i<b.length;i++)
{
    c = a[i];
    a[i] = b[i];
    b[i] = c;
}
}
```

b) (10 points) Write a method **PrintArray** that takes a 2D integer array as a parameter and prints the lower left triangle elements of the array. You can assume that the 2D array has the same number of rows as columns. The method doesn't return anything.

For example, for the array

```
1 2 5
3 7 5
6 1 2 ,
```

The output should be

```
1
3 7
6 1 2
```

```
public void PrintArray(int[][] a)
{
    for(int i=0;i<a.length;i++)
    {
        for(int j=0;j<=i;j++)
        {
            System.out.print(a[i][j]+" ");
        }
        System.out.println();
    }
}
```

2. (15 points)

Write a method that takes an array of characters as a parameter and determines if each character in the array is a vowel (a, e, i, o, u) or a consonant using one or more switch statements. (You can assume all characters are lower case for this problem). Your method should print out each value in the array and also print out whether it is a vowel (print a 'V') or a consonant (print a 'C'). *You can only use while loops in your method* (no for or do loops allowed for this problem).

```
Public void method1(char[] array)
{
int i=0;
while (i<array.length)
{
switch(array[i])
{
    case 'a':
    case 'e':
    case 'i':
    case 'o':
    case 'u':
        System.out.println(array[i]+":V");
        break;
    default: System.out.println(array[i]+":C");
}
i++;
}
```

3. (22 points)

Write a class **Problem3** that contains a 2D int array, **arr**, which has the same number of rows as columns.

- a) (8 points) Write a constructor that takes the number of rows and columns of the array as parameters, creates the array, and initializes each element of the array to a random int between 2 and 9. Also define any class variables for the class and write out the class header.

```
import java.util.*
Public class Problem3
{
    int[][] arr;
    Random rand = new Random();
    Public Problem3(int rows, int cols)
    {
        arr = new int[rows][cols];
        for(int i=0;i<rows;i++)
            for(int j=0;j<cols;j++)
                arr[i][j] = rand.nextInt(8)+2;
    }
}
```

- b) (10 points) Write a method **FindSum** that calculates the sum of the elements along both the diagonals of the array and returns the maximum of the two sums. The method takes a 2D int array as a parameter. Your method should work for a 2D array of any size that has the same number of rows and columns. Examples of the diagonals of a 3x3 array are:

1	2	3	->	1	and	3
5	6	7		6		6
8	9	0		0		8

```

public int FindSum(int[][] a)
{
    int sum = 0, sum1 = 0, length;
    length = a.length;
    for(int i=0; i<length; i++)
    {
        sum = sum + a[i][i];
        sum1 = sum1 + a[length-i-1][i];
    }
    if(sum > sum1)
        return sum
    else
        return sum1
}

```

- c) (4 points) Write a main method that creates an object of the class **Problem3**, calls the method **FindSum** and prints the value returned by the **FindSum** method.

```

Public static void main(String args[])
{
    Problem3 obj = new Problem3 (5,6);
    int sum = obj.FindSum(obj.arr);
    System.out.println(sum);
}

```

4. (15 points )

Class **Problem4** has the following characteristics:

- a) (8 points) Method **ReadArray** takes no parameters. The method reads 35 integers from the keyboard and stores them in an integer array. A reference to the array is returned by the method. Write method **ReadArray** below along with the class header and any **import** library calls.

```
import java.util.Scanner;
public int[] ReadArray()
{
    int[] arr = new int[35];
    Scanner scan = new Scanner(System.in);
    for(int i=0;i<35;i++)
    {
        arr[i] = scan.nextInt();
    }
    return arr;
}
```

- b) (7 points) The **main** method for **Problem4** creates an array using **ReadArray**. The values located at the even indices (e.g. 0, 2, 4, etc) in the arrays are then printed out *in reverse order* (e.g. the value located at the last even index in the array is printed first, etc). Write the **main** method for **Problem4** below.

```
public static void main(String args[])
{
    Problem4 object1 = new Problem4();
    int[] array = new int[35];
    int length;
    array = object1.ReadArray();
    if((array.length-1)%2==0)
```

```

        length = (array.length-1);
    else
        length = (array.length-2);
    for(int i = length ; i>=0; i=i-2)
    {
        System.out.println(array[i]);
    }
}

```

5) (13 points) Write a method **CompareInt** which takes in two 2D arrays of integers (**array1** and **array2**) as parameters. The method compares the value of each entry in the arrays at the same (x, y) location and the result is stored in a 2D Boolean array (**boolarray**) created by the method. If the value in the (x, y) position of **array1** is evenly divisible by the value in the (x, y) position of **array2**, true is stored in the (x, y) position of **boolarray**, otherwise false is stored. *Note: evenly divisible means no remainder is created when the first value is divided by the second value.*

Your method should assume that all arrays are the same size. The boolean array is returned by the method.

```

public Boolean[][] CompareInt(int[][] array1, int[][] array2)
{
    Boolean[][] boolarray = new Boolean[array1.length][array1[0].length];
    for(int i=0;i<array1.length;i++)
    {
        for(int j=0;j<array1[i].length;j++)
        {
            if(array1[i][j]%array2[i][j] == 0)
                boolarray[i][j] = true;
            else
                boolarray[i][j] = false;
        }
    }
}

```

```
}  
return boolarray;  
}
```

6) (15 points)

a) (3 points) Write two characteristics of a static method that are unique.

1) **Can be invoked through the class name without having to create an object of the class**

2) **Cannot reference non static variables which exist only in an instance of the class.**

b) (4 points) What is printed out by this code? Include any run-time error messages that may be printed out.

```
char[] array = {'a', 'b', 'c', 'd'};  
for (int i=0; i<=4; i++)  
{  
    System.out.println(array[i]);  
}
```

**a**

**b**

**c**

**d**

**Array index Out of bound exception**

c) (4 points) What is the output of the following piece of code?

```
for( int i= 27; i>1 ; i = (i%4))
```

```
{
```

```
    System.out.println(i);
```

```
}
```

**27**

**4**

d) (4 points) Write a statement to create an instance of **ArrayList**. Invoke the add method and add a string “RT” to the array list.

```
ArrayList arr = new ArrayList();
```

```
arr.add(new String(“RT”));
```