Name: Key UTA ID:

Instructions:

1. The test is worth 100 points. The point value of each question is given with the question. There are also extra credit questions at the end.
2. Read all of the instructions for each question and answer what is asked. Do not write down random stuff if you don’t know the answer.
3. All questions that have the same question number are related to each other. However, they are not all necessarily dependent on each other so you can skip around if needed.
4. The test is open book and open notes for all printed and hand-written material. You may NOT bring an electronic book or any electronic device to use during the test (no computer, no smart phone, etc.) You may use as much printed or written material as desired including copies of code examples.
5. You will write your answers on the test pages. If additional space is needed, you may use the backs or bottoms of the pages or extra paper. Please make a note on the test page whenever your answer continues onto another page and indicate where the answer is.
6. Please write legibly and large enough to read easily. Your writing should be readable if the test is sitting on a desk in front of me. I am not looking for perfect handwriting but it does need to be legible. I will deduct points if your answers are much more difficult to read than those of the general student.
7. For multiple choice questions, circle the letter of your answer choice. Circle only one choice for your answer.
8. If you have a question during the test, please raise your hand. The TA and I will be available to come hear your question. Sometimes we may not be able to answer your question because it gives you too much information but you should always ask.
9. You have 2 hours and 30 minutes to complete the test.

1.a) Is the error likely to be in the right-hand side of the assignment on line 234 or the left-hand side? Tell why you chose your answer. {6 points}

**The error will be on the left hand side because that is where the array access is. You can’t have an out of bounds index unless there is an index.**

1.b)

**B) run-time exception** {4 points}

1.c)

**196 int adjLtrs = adjectives[0].length - 1; OR**  {8 points}

**227 for (int n = (adjLtrs-1); n > 0; n--) {**

1.d) Is the error a (Circle the letter of your choice) {4 points}

**C) run-time logic error**

1.e) If only one line needs to change in order to correct this error, what line would you change and how would you change it? Write the line number and write the whole corrected line.

**227 for (int n = adjLtrs; n => 0; n--) {**  {8 points}

1.f) **letter by letter, 229** {6 points}

1.g) What does the variable numLtrs tell you? Circle the letter of all the things below that are true. {4 points}

**A) The length of a line from the file, assuming *inFIle* is connected to the input file.**

**B) The number of letters in the current string that was read in.**

C) The length of a string stored in the 2D array.

**D) The location of the char that will be right after end of the word’s letters in the 2D array.**

1.h) In the for loop at line 227 (as written), what is the starting value of n, what is the ending value of n, and how many array elements would that cover? If needed, give answer as equation.

Starting value of n is : **adjLtrs** {8 points}

Ending value of n is :  **1**

The number of array elements covered is: **adjLtrs – 1 + 1 = adjLtrs**

2.a) {8 points}

*recur(2,6,0)* will generate a total of **3** calls // corrected this to recur(2,6,0)

*recur(1,8,0)* will generate a total of **5** calls // (1, 8, 0)

*recur(1,5,0)* will generate a total of  **3** calls // (1, 5, 0)

*recur(3,10,0)* will generate a total of  **5** calls // 3, 10, 0

2.b) “Yes” or “No” and explain why you can or cannot generalize. {8 points}

**Yes, with input (*limita, limitb,*0) you can generalize the number of recursive calls to**

 **Math.ceil( ( *limitb* – *limita* ) / 2.0 ) + 1 ) or ( *limitb* – *limita* ) + 1 rounded up**

 **2.0**

**“The ceiling function of (one plus the ((difference of limitb minus limita) over two))”**

3. Convert the numbers below to the appropriate alternate base. {12 points}

12310 would convert to **1111011** 2

123 – 64 = 59 - 32 = 27 - 16 = 11 - 8 = 3 - 2 = 1 - 1 = 0

 1 1 1 1 0(4) 1 1

101102 would convert to **22** 10

(1 x 16) + (0 x 8) + (1 x 4) + (1 x 2) + (0 x 1) = 16 + 4 + 2 = 22

25710 would convert to 1 0000 00012 =  **101** 16

257 – 256 = 1 - 1 = 0

 1 0(128) 0(64) 0(32) 0(16) 0(8) 0(4) 0(2) 1

4. Explain what the Java statement below is doing {4 points}

int roll = (int)(Math.random() \* sidesOfACube);

**This simulates the roll of a six-sided dice**

5. Arrows diagram

XC1) specifically to catch errors before they occur? Circle the letters of all that apply. {4 points}

**A) Printing data values as the program runs**

**B) Putting statements in a try structure**

**C) Setting a boolean flag based on a condition**

XC2) Which of the following is NOT true about ArrayLists? Circle the letter {2 points}

**B) An ArrayList can shrink to fit its data but cannot expand**

XC3) The Java API does NOT include: Circle the letter of your answer {2 points}

**A) Netbeans, Eclipse and other programming environments**

XC4) What would your response be to me saying

“Have a fun and safe winter break! Thank you for being a student in my class this semester. Enjoy your celebrations and refresh yourself for next semester.” ?

 {Any polite answer will receive 2 points.}

1. Using the code below answer the questions that follow. This code is part of a method to take words that are adjectives from a file (*inFile*) as Strings (*modifier*) and store each word into a two-dimensional character array (*adjectives*). Assume that array *adjectives*, the counter *i,* and the Scanner *inFile* have been declared and initialized.

196 int adjLtrs = adjectives[0].length; // Assume that array adjectives is defined and is valid

223 String modifier = inFile.nextLine(); // Assume that Scanner inFile is defined and is valid

224 int numLtrs = modifier.length();

----

227 for (int n = adjLtrs; n > 0; n--) {

228 if (n == numLtrs) {

229 adjectives[i][n] = modifier.charAt(n); // Assume that i is defined and is valid

230 System.out.println("DEBUG: adjectives["+i+"]["+n+"] is "+adjectives[i][n]);

231 numLtrs--;

232 }

233 else {

234 adjectives[i][n] = ' '; // Assume that i is defined and is valid

235 }

236 }

 The code above gives an ArrayIndexOutOfBoundsException on line 234.

1.a) Is the error likely to be in the right-hand side of the assignment on line 234 or the left-hand side? Tell why you chose your answer. {6 points}

The error will be on the left hand side because that is where the array access is. You can’t have an out of bounds index unless there is an index.

1.b) Is the error a (Circle the letter of your choice) {4 points}

A) compile-time syntax error

B) run-time exception

C) run-time logic error

1.c) If only one line needs to change in order to correct this exception, what line would you change and how would you change it? Write the line number and write the whole corrected line. NOTE: There are at least two possible answers here, if not more, that are correct. {8 points}

196 int adjLtrs = adjectives[0].length - 1; OR

227 for (int n = (adjLtrs-1); n > 0; n--) {

Assume now that the out of bounds exception is fixed. Now the method runs but the adjectives in the array include “ntelligent”, “appy”, and “xcited”.

1.d) Is the error a (Circle the letter of your choice) {4 points}

A) compile-time syntax error

B) run-time exception

C) run-time logic error

1.e) If only one line needs to change in order to correct this error, what line would you change and how would you change it? Write the line number and write the whole corrected line.

 {8 points}

227 for (int n = adjLtrs; n => 0; n--) {

1.f) Is this code saving the whole word into the 2D char array in one statement or is it saving the word letter by letter? Which line in the code is actually putting the word into the array? Answer with “whole word” or “letter by letter” and give the line number.

 {6 points}

letter by letter, 229

1.g) What does the variable numLtrs tell you? Circle the letter of all the things below that are true. {4 points}

A) The length of a line from the file, assuming *inFIle* is connected to the input file.

B) The number of letters in the current string that was read in.

C) The length of a string stored in the 2D array.

D) The location of the char that will be right after the end of the word’s letters in the 2D array.

1.h) In the for loop at line 227 (as written), what is the starting value of n, what is the ending value of n, and how many array elements would that cover? (Ex. If it started at 2 and went to 5, then that would cover the elements at locations 2, 3, 4, and 5 which would be 4 elements.) If needed, give your answer as an equation. {8 points}

Starting value of n is : adjLtrs

Ending value of n is : 1

The number of array elements covered is: adjLtrs – 1 + 1 = adjLtrs

227 for (int n = adjLtrs; n > 0; n--) {

2) Given the recursive method below answer the questions that follow:

 public static int recur(int limita, int limitb, int val)

 {

 if (limita == limitb)

 {

 val = val + limita;

 }

 else if (limita < limitb)

 {

 val = recur(limita+1, limitb-1, val) + limita + limitb;

 }

 // else limita > limitb - do nothing

 return val;

 }

2.a) If the method *recur* is called with the various calls below, how many times will method *recur* be called in total including the first call for each statement? Answer with a number of calls for each call to *recur* given below. {8 points}

*recur(2,6,0)* will generate a total of 3 calls // corrected this to recur(2,6,0)

*recur(1,8,0)* will generate a total of 5 calls // (1, 8, 0)

*recur(1,5,0)* will generate a total of 3 calls // (1, 5, 0)

*recur(3,10,0)* will generate a total of 5 calls // 3, 10, 0

2.b) Given your answers above, can you generalize how many times the recursive function is called in relation to the given inputs? Answer “Yes” or “No” and explain why you can or cannot generalize. {8 points}

Yes, with input (*limita, limitb,*0) you can generalize the number of recursive calls to

 Math.ceil( ( *limitb* – *limita* ) / 2.0 ) + 1 ) or ( *limitb* – *limita* ) + 1 rounded up

 2.0

“The ceiling function of (one plus the ((difference of limitb minus limita) over two))”

3. Convert the numbers below to the appropriate alternate base. Remember that a subscript indicates the base for the number. Show your work for partial credit. {12 points}

12310 would convert to 1111011 2

123 – 64 = 59 - 32 = 27 - 16 = 11 - 8 = 3 - 2 = 1 - 1 = 0

 1 1 1 1 0(4) 1 1

101102 would convert to 22 10

(1 x 16) + (0 x 8) + (1 x 4) + (1 x 2) + (0 x 1) = 16 + 4 + 2 = 22

25710 would convert to 1 0000 00012 = 101 16

257 – 256 = 1 - 1 = 0

 1 0(128) 0(64) 0(32) 0(16) 0(8) 0(4) 0(2) 1

4. Explain what the Java statement below is doing in terms of its purpose not just writing the Java in English instead of code, i.e. what action is this statement simulating? Base your answer on the names of the variables, the method calls, and the structure of the statement. {4 points}

int roll = (int)(Math.random() \* sidesOfACube);

This simulates the roll of a six-sided dice

5. On the left are some Java statements (some are partial statements). On the right are choices of control structure categories. Match the elements on the left with the choices on the right. Show your match by drawing an arrow from the end of the statement on the left to the matching choice on the right. Multiple elements will match with the same choice but each element will only match with one choice. {20 points}

NOTE: Assume that all variables are declared and the tests are valid. Also assume that correct code will follow any open curly brace “{“ so there are no syntax errors.

if ( num > limit ) {

while ( count != 0) { **Sequential control structure**

return fac(k-1) \* k; // in method *fac*

limit = 100;

try{ // with appropriate catch block

 **Repetition control structure**

for (int i = 0; i < limit; i++) {

int count;

switch(count) {

do { // with matching while **Selection control structure**

System.out.println(“ Number is: “+num);

Extra credit

XC1) Which of the following could be part of a programming strategy specifically to catch errors before they occur? Circle the letters of all that apply. {4 points}

A) Printing data values as the program runs

B) Putting statements in a try structure

C) Setting a boolean flag based on a condition

D) Using a for loop to iterate through an array

XC2) Which of the following is NOT true about ArrayLists? Circle the letter {2 points}

A) The angle brackets surround the data type to be stored in the ArrayList

B) An ArrayList can shrink to fit its data but cannot expand

C) The primitive data types cannot be used with an ArrayList

D) An ArrayList must be initialized before it can be used

XC3) The Java API does NOT include: Circle the letter of your answer {2 points}

A) Netbeans, Eclipse and other programming environments

B) Classes that are needed to do input and output in Java programs

C) Definitions of classes with special purposes such as the Math class and the Random class

D) Online documentation of its contents

XC4) What would your response be to me saying

“Have a fun and safe winter break! Thank you for being a student in my class this semester. Enjoy your celebrations and refresh yourself for next semester.” ?

 {Any polite answer will receive 2 points.}