CSE 1321 Final Exam Random

Rutvik Dhira Marakana

TOTAL POINTS

93 / 100

QUESTION 1

1 Methods - Hunger Games 17 / 20

✓ + 4 pts METHOD keyword used

✓ + 2 pts END METHOD used

✓ + 5 pts Two parameters passed in

✓ + 6 pts Math for calculating the correct value

✓ + 3 pts RETURN a single numerical value for pounds gained

- 2 pts Total weight is mostly calculated correctly, but there is a minor math issue

- 3 pts Total weight is mostly calculated correctly, but there were math issues

- 4 pts A value for total weight is calculated, but there were substantial math issues

- 2 pts Did not use values passed in as parameters

+ 0 pts Blank or not credit awarded

- 3 Point adjustment

Can't have two return type (-2) Logic wrong for cal < 2500

QUESTION 2

2 Methods - Part 2 10 / 10

✓ + 4 pts Called method by name

✓ + 4 pts Passed two parameters

✓ + 1 pts Printed out the results, either directly or by storing in a variable

✓ + 1 pts Called the method twice with different values

+ 0 pts Blank or no credit

- 1 pts Variables are used, but not initialized

- 2 pts Variables are not declared or initialized

- 2 pts Should use class name to call method

QUESTION 3

3 Repetition - Papers Please 20 / 25

+ 2 pts Declared userName variable for user input correctly.

+ 2 pts Declared password variable for user input correctly.

+ 1 pts Declared attempts/count (outside loop) correctly.

✓ + 5 pts Used a WHILE or DO-WHILE Loop correctly

✓ + 5 pts Set WHILE/Sentinel Values conditions correctly

✓ + 2.5 pts Incremented Attempts (or equivalent) correctly

✓ + 2.5 pts Read User Input properly (Username & Password, inside a single loop)

✓ + 2.5 pts Broke loop if attempts were >= 10 or correct info was entered (or criteria was meet otherwise).

✓ + 2.5 pts Printed statements correctly

+ 0 pts Incorrect/unattempted

QUESTION 4

4 1D - Triple Threat 15 / 15

✓ + 1 pts Used a FOR Loop

✓ + 1 pts Set initial index properly (default is 0 but may differ depend on code).

✓ + 2 pts Set FOR Loop length correctly (9997 or bob.length-3, since index starts at 0 and you want to stop at bob{i+2}, may differ depending on code)

✓ + 1 pts Incremented by 1

✓ + 5 pts Set conditions for consecutive 0s correctly. E.g: bob[i] == 0, bob[i+1] == 0, bob[i+2} == 0 (not a number containing three 0s in a row but three individual zeros in a row).

✓ + 5 pts Printed statement correctly.

+ 0 pts Incorrect/Unattempted

QUESTION 5

5 Sort Differentiation 5 / 5

✓ + 5 pts **Correct**

+ 0 pts Incorrect

QUESTION 6

6 OOP - The Most Boring Class Alive 25 / 25

✓ + 2 pts **CLASS keyword BEGIN**

✓ + 4 pts **Attributes/variables declared**

✓ + 3 pts **CONSTRUCTOR keyword**

✓ + 3 pts **Appropriate pairing of parameters to variables in constructor**

✓ + 3 pts **Attributes initialized in constructor**

✓ + 1 pts **END constructor**

+ 1 pts Area and Perimeter methods declared, but missing ( ).

✓ + 1 pts **Area method properly declared**

✓ + 2 pts **Area method returns the area of a rectangle**

+ 0 pts Incorrect calculation for Area

✓ + 1 pts **END Area method**

✓ + 1 pts **Perimeter method properly declared**

✓ + 2 pts **Perimeter method returns the perimeter of a rectangle**

+ 0 pts Improper calculation for Perimeter

✓ + 1 pts **END Perimeter method**

✓ + 1 pts **END CLASS**

+ 0 pts No rubric points available

QUESTION 7

7 Extra Credit Drawing 1 / 0

✓ + 1 pts **Correct**

+ 0 pts Blank

CSE 1321 Lecture Final Exam Cover Sheet

1) **Print your Name, ID# and NetID on each page.**
2) **THERE ARE SIX (6) QUESTIONS AND AN EXTRA CREDIT QUESTION ON THIS EXAM. CHECK EACH PAGE TO MAKE SURE YOU HAVE ALL QUESTIONS!**
3) Student has 110 minutes to complete the exam
4) Student MAY NOT use notes or book
5) **Please make sure to check the corresponding language box for each question:**
   a. Pseudocode answers should be clearly explained enough that a reader could take the answer and turn it into source code with minimal interpretation.
   b. Source code must be exact source code (include all required symbols, syntax, and indentation). It should be written to where a compiler would allow that code to run without any changes from the reader.
6) Student is not allowed any electronic devices that can be used to look up or store answers.
7) All answers are to be your own, without the assistance of others
8) Partial credit will be given where appropriate

Student Name: Rutwik Mavakumar

Student KSU ID# 000842768

Student NetID: smavakum

Student Signature: Rutwik

Date: 12/7/18

Student Name: Rutvik Morakama NetID: rmorakam KSU ID# 00084247668

Q1 (20 pts): Methods – <u>Hunger Games:</u> Assume that the average person can consume 2500 calories in one day without gaining any weight. There are 3500 calories in one pound of fat. Your job is to write a function/method that takes in two parameters: 1) the number of calories per day that a person eats and 2) the number of years a person will eat that many calories. The program will <u>return</u> the number of pounds the person will gain (and technically work for losing weight as well). Yes, there are 365 days in one year! Note: a person that eats 2500 calories a day will not gain weight (for this question). Why?

Answer is in Pseudocode: ☑ C#: ☐ Java: ☐ Python: ☐

```
METHOD hungerGames(int cal, int yr)
BEGIN
  CREATE int weightgain
  IF (cal ≥ 2500) THEN
        weightgain ← (cal - 2500)/3500
  END IF
  CREATE int totalweightgain
  CREATE int totalweightloss
  CREATE int weightloss
  IF (cal ≤ 2500) THEN
     weightloss ← (2500 - cal)/3500
  END IF
  totalweightgain ← weightgain * 365 * yr
  totalweightloss ← weightloss * 365 * yr
```

Student Name: Rutvik Marakana NetID: rmarakan KSU ID# 000844768

Continue answer for Question 1 here if necessary:

RETURN totalweightloss
RETURN totalweightgain
END METHOD

Q2 (10 pts): Methods Part 2: Correctly call the method in Q2 two times and print the results.

Answer is in Pseudocode: ☐ C#: ☐ Java: ☑ Python: ☐

```
int a=0;
a=hungerGames(6000,1);
System.out.println("The person will gain "+a+" pounds");
a = hungerGames(12000,1);
System.out.println("The person will gain "+a+" pounds");
```

Student Name: Rutvik Mandava NetID: rmandava KSU ID# 000844768

Q3 (25 pts): Repetition – Papers, Please? One of the most frustrating things about technology is forgetting your password, incorrectly guessing it multiple times, and then being locked out of your account. Using either a WHILE or DO-WHILE loop (only), write a program that will repeatedly ask the user for his/her name and password, and then print "Welcome!" upon a successful login. However, after 10 incorrect attempts, they should not receive a greeting, but instead the message "See system administrator to reset password, sucker!" For this question, accept only one account name: "jbub" whose sickening password is "SparklingPuppies".

Answer is in Pseudocode: ☐ C#: ☐ Java: ☐ Python: ☐

```
boolean a ← false
DO
    READ user_input for account name and
    password.
    acc_name ← user_input name
    pass ← user_input password
    n ← 1
    IF ( name ← "jbub" && pass ← "SparklingPuppies"
        && n <= 10) THEN
        PRINT "welcome!"
        BREAK
    ELSE
        a ← true
    END IF
    n ← n + 1
    IF ( n == 11)
        PRINT "See System administrator to reset
        password, sucker!"
        BREAK
    END IF
WHILE (a == true)
END WHILE
```

Student Name: Rutvik Maraboina NetID: rmaraboin KSU ID# 000844768

Q4 (15 pts): 1D – Triple Threat! Imagine we have an array (called "bob") of 10,000 random numbers that will be fed into a program. However, if there are three zeroes in a row (000), the program will break, the computer will smoke, and life as we know it will cease to exist. Write code that scans array *bob* and detects if three zeroes appear in a row. If three zeroes do appear in a row, print "SMOKE! RUN!". Otherwise, the program should not output anything. Hint: start by finding the first zero. What should you check for next?

Answer is in Pseudocode: ☐ C#: ☐ Java: ☑ Python: ☐

```
boolean a = false;
for (int i = 0; i <= bob.length-2; i++){
    if (bob[i] == 0 && bob[i+1] == 0
        && bob[i+2] == 0)
    {
      a = true;
      break;
    }
}
if (a == true)
{
  System.out.println("SMOKE! RUN!");
}
```

Q5 (5 pts): The code below is the closest implementation of which of the following sorts:

A) BubbleSort
B) SelectionSort
C) None of the above

Answer is (write large and legible):

A
or
Bubble Sort

```
CREATE numbers[7]

numbers ← { 88, 71, 47, 93, 62, 15, 49 }

CREATE temp


FOR i ← 0 to 6

  FOR j ← 0 to 6 - (i-1)

        IF (numbers[j + 1] < numbers[j]) THEN

              temp = numbers[j]

              numbers[j] = numbers[j + 1]

              numbers[j + 1] = temp

        END IF

  END FOR

END FOR
```

Student Name: Rutvika Mamidanna NetID: rmamidan KSU ID# 008447768

Q6 (25 pts): OOP – The Most Boring Class Alive. Almost all programming textbooks insist on including a Rectangle class example that includes two class variables - width and height. It also includes three methods inside the class – a constructor (that takes in a width and height as parameters), Area (that returns the area of the rectangle), and Perimeter (which returns the perimeter of the rectangle). Write class Rectangle, including its class variables and methods, and try not to fall asleep. Must… stay… awake….zzzzzzz

Answer is in Pseudocode: ☑ C#: ☐ Java: ☐ Python: ☐

```
CLASS Rectangle
BEGIN
    CREATE int width
    CREATE int height

    CONSTRUCTOR Rectangle (int w, int h)
    BEGIN
        width ← w
        height ← h
    END CONSTRUCTOR()

    METHOD Area()
    BEGIN
        RETURN THIS.width * THIS.height
    END Area()

    METHOD Perimeter()
    BEGIN
        RETURN 2*(THIS.width + THIS.height)
    END Perimeter()
END CLASS
```

Student Name: Rutvik Morakama NetID: rmorakam KSU ID# 000844768

Extra credit (+1 pt): Draw something (except the word something).