Practice Final

2024-09-19

Contents

The final exam will be a closed-book paper exam without a calculator. Exam questions will be similar in type to those found here, but fewer in number. While this practice exam is a good study guide, we highly recommend being familiar with *all the material* (including but not limited to your previous exams, labs, projects, quizzes and homework) as well.

Problem 0 (Warm-up)

Solutions for those exercises.[1]

1. What is the escape sequence for a new line?
2. What type is the result of `8 * 12M`?
3. What is the return type of a constructor?
4. What operator would you use to see if int a and int b are equal?
5. List 4 datatypes.

[1] https:/princomp.github.io/exercises_w_sol/past/practice_final

6. List 4 reserved words (keywords).
7. What is the difference between a variable and a constant?
8. Write a statement that declares a constant of type int named `DaysInWeek` and sets its value to 7.
9. In an exam class, if I want to keep track of the total number of exams should the attribute be static or non-static?
10. What operator is used to find out the remainder from division?
11. Write a condition that evaluates to true if an int length is between 4 and 16, both inclusive.
12. How many times would a for loop with this header run? `for(int i=5;i<12; i++)`
13. Write a statement or statements that creates an int array of size 50 with each index containing that index as its value. (i.e. 0 at `[0]`, 13 at `[13]`, 49 at `[49]`, etc.).
14. Write a statement or statements to create a random number generator called `examRand` and use it to generate a random number between 40 and 57 (inclusive).

Problem 1

Solutions for those exercises.[2]

Consider the code below:

```
class VirtualPet{
    private string name = "Blank";          // Name of the
     ↪  pet.
    private decimal hungerLevel = 1m;        // Level of
     ↪  hunger, with 1 being full, in percent.
    private decimal happinessLevel = 1m;     // Level of
     ↪  happiness, in percent

    public void SetName(string nameP)
    {
        name = nameP;
    }
}
```

1. Write a statement to instantiate a `VirtualPet` object called `firstPet`.

2. Write a getter for the name attribute.

3. Write a statement that would display to the screen the name of the `firstPet` object you created previously. What would be displayed?

[2] https:/princomp.github.io/exercises_w_sol/past/practice_final

4. Write a setter for the `hungerLevel` attribute that takes one decimal. The argument should be assigned to the `hungerLevel` attribute only if it is between 0 and 1 (both included), otherwise the attribute should get the value 0.
5. Draw the UML diagram for the `VirtualPet` class, including the methods you just added.
6. Write a constructor that takes 3 arguments (`string`, `decimal`, `decimal`) for the `VirtualPet` class. Your constructor should be such that if one of the decimal arguments is not between 0 and 1 (both included), then 0 gets assigned to both decimal attributes.
7. Your earlier statement that created the firstPet object will no longer compile after you add the constructor. Why is this the case?
8. Write a statement that would create a new `VirtualPet` object called `secondPet` using the constructor you just added (the argument values are up to you).
9. Write a `ToString` method for the `VirtualPet` class. It should display the name, `hungerLevel`, and `happinessLevel`. (Bonus) Display `hungerLevel` and `happinessLevel` graphically: for instance, if `hungerLevel` is at 4.5, display "Hunger: XXXX". You may freely use symbols as if they were normal letters.
10. Write a statement that would use the `ToString` method from the `VirtualPet` class you just added to display information about the `secondPet` object.

Problem 2

Solutions for those exercises.[3]

This question will have you partially design, implement and use class to represent hamburgers. A Burger has a name, a price, a Boolean for dairy, and a type (typically beef, pork, chicken, veggie).

1. Draw the UML diagram for the Burger class, assuming it contains the listed attributes, a getter for the name attribute and a setter for the price attribute. Do not include any other methods.
2. Write a getter for the name attribute.
3. Write a setter for the price attribute.
4. Write a constructor that takes 4 arguments and sets the value of the attributes to be the value of the arguments.

[3] https:/princomp.github.io/exercises_w_sol/past/practice_final

5. Write an additional constructor that takes a name, a dairy, and a type. The price should then be set according to the following table. If the value for type is not in the table, price should be set to -99.99.

6. Write a static method Promotion that takes as an argument a price and returns a value 75% of the argument.

7. Write a `ToString` method. The string returned should contain the values of all attributes.

8. Write a statement/statements that:

- Displays the result of passing 12.84 to Promotion.
- Instantiates a Burger object named OldBeefy with the values “Old Beefy”, 1.99, true, and “beef”.
- Changes the price of OldBeefy to 2.29.
- Displays the name (and only the name) of OldBeefy.
- Store the value returned by calling the ToString method with OldBeefy in a variable.

Problem 3

Solutions for those exercises.[4]

Complete the table based on the code.

x	y	z	Displays
-1	‘e’	18.2M	
-1	‘a’	-2	
0	‘c’	4.6M	
1	‘d’	2	
-1	‘b’	115	
1	‘d’	-33.7M	
0	‘a’	0	
1	‘c’	13	
			5

```
int x;
char y;
decimal z;

// x, y, and z are given legal values

if(x<0 && y == 'a'){
```

[4] https:/princomp.github.io/exercises_w_sol/past/practice_final

```
  Console.Write("1");
}
else if(z%2==0){
  Console.Write("2");
}
else if(y=='c' || y=='d'){
  Console.Write("3");
}
else if(x!=0 && z!=0){
  Console.Write("4");
}
else{
  Console.Write("5");
}
```

Problem 4

Solutions for those exercises.[5]

Given two int arrays of equal length, write a code segment that compares the values at each index to see if they match. Return the total number of matches.

Problem 5 (Deceptively hard)

Solutions for those exercises.[6]

Given two string arrays (array A and array B) of unknown (possibly different) lengths, determine if there are any values found in both A and B. If they exist, display them to the screen. At the end of the program, display the total number of common values between A and B. If there are repeating values in either or both arrays, each should only be counted once.

(Bonus): How could Lists be used to make this problem easier?

Problem 6

Solutions for those exercises.[7]

[5] https:/princomp.github.io/exercises_w_sol/past/practice_final

[6] https:/princomp.github.io/exercises_w_sol/past/practice_final

[7] https:/princomp.github.io/exercises_w_sol/past/practice_final

Write a program that declares an int variable called "pin" and asks the user for their pin. As long as the user enters something that is not a number, is negative, or greater than 9999, your program should ask again.

(Bonus): Your code should make sure that the pin has exactly 4 digits, including leading zeros.

Problem 7

Solutions for those exercises.[8]

1. Write a statement that would create an int array of size 100.
2. Write a series of statements that would ask the user to enter a value for each cell in the array (no need to perform user-input validation, but you may if you like).
3. Write a series of statements that would ask the user to enter a value, displaying "In your array" if the value is in your array.
4. Write a series of statements that would display the sum of values in the array.
5. Write a series of statements that would display the product of all the non-zero values in the array.
6. Write a series of statements that would display the smallest index of the greatest value in the array.

[8] https:/princomp.github.io/exercises_w_sol/past/practice_final