Exam #2

Thursday, September 2, 2021

- This exam has 6 questions, with 100 points total.

- **You should submit your answers in the Gradescope platform (not on Brightspace).**

- You have **two hours**.

- **It is your responsibility to take the time for the exam** (You may use a physical timer, or an online timer: https://vclock.com/set-timer-for-2-hours/).
  Make sure to upload the files with your answers to gradescope BEFORE the time is up, while still being monitored by ProctorU.
  We will not accept any late submissions.

- In total, you should upload 3 '.cpp' files:
  - One '.cpp' file for questions 1-4.
    Write your answer as one long comment (/* … */).
    Name this file 'YourNetID_q1to4.cpp'.
  - One '.cpp' file for question 5, containing your code.
    Name this file 'YourNetID_q5.cpp'.
  - One '.cpp' file for question 6, containing your code.
    Name this file 'YourNetID_q6.cpp'.

- **Write your name, and netID at the head of each file.**

- This is a closed-book exam. However, you are allowed to use:
  - CLion or Visual-Studio. You should create a new project and work ONLY in it.
  - Two sheets of scratch paper.
  - Scientific Calculator

  Besides that, no additional resources (of any form) are allowed.

- You are not allowed to use C++ syntactic features that were not covered in the Bridge program so far.

- Read every question completely before answering it.
  Note that there are 2 programming problems at the end**.**
  Be sure to allow enough time for these questions

Part I – Theoretical:

- *You should submit your answers to all questions in this part (questions 1-4) in **one** '.cpp' file. Write your answers as one long comment (/* … */).*
  Name this file 'YourNetID_q1to4.cpp'.

- *For questions in this part, try to find a way to use regular symbols.*
  For example, instead of writing a^b you could write a^b, instead of writing $\Theta(n)$, you could write theta(n), instead of writing $\binom{n}{k}$ you could write C(n, k), etc.
  Alternatively, you could also make a note, at the beginning of your answer, stating what symbol you used to indicate a specific mathematical notation.

Question 1 (14 points)
Use mathematical induction to prove that for any positive integer $n \geq 2$, $3^n > 2^n + n^2$.

Question 2 (12 points)
In how many permutations of the digits 1-9, there are only even digits in the first 3 places?
For example,
- In **4**, **2**, **8**, 3, 1, 7, 5, 9, 6 there are only even digits in the first 3 places.
- In **4**, **1**, **8**, 3, 2, 7, 5, 9, 6 the first 3 places are not all even digits.

Explain your answer.

Question 3 (15 points)
A 5-length bit string is randomly generated.
Let *X* be the random variable that is equal to the number of 1s in the bit string.
For example, for the bit string 01101, *X* would be 3 (as there are 3 1s in 01101).
Note: Assume that all 5-length bit strings are equally likely to be generated.

a. Find the distribution of *X*. That is, for each possible value of *X*, say what is the probability *X* would get that value.
b. What is *E(X)*? That is, find the expected value of *X*.

Explain your answers.

Question 4 (14 points)
Analyze its running time of **`func1`** and **`func2`**.
Explain your answers.

Note: Give your answers in terms of asymptotic order. That is, $T(n) = \Theta(n^2)$, or $T(n) = \Theta(\sqrt{n})$, etc.

```
int func1(int n){
    int* arr;
    int i, j, count;

    arr = new int[n];
    arr[0] = 1;
    for(i = 1; i < n; i += 1)
        arr[i] = 1 + arr[i-1];

    count = 0;
    for(i = 0; i < n; i += 1) {
        for (j = 1; j <= arr[i]; j += 1)
            count += 1;
    }

    return count;
}
```

```
int func2(int n){
    int* arr;
    int i, j, count;

    arr = new int[n];
    arr[0] = 1;
    for(i = 1; i < n; i += 1)
        arr[i] = 2 * arr[i-1];

    count = 0;
    for(i = 0; i < n; i += 1) {
        for (j = 1; j <= arr[i]; j += 1)
            count += 1;
    }

    return count;
}
```

Part II – Coding:

- *Each question in this part (questions 5-6), should be submitted as a '.cpp' file.*
- *Pay special attention to the style of your code. Indent your code correctly, choose meaningful names for your variables, define constants where needed, choose most suitable control statements, etc.*
- *In all questions, you may assume that the user enters inputs as they are asked. For example, if the program expects a positive integer, you may assume that user will enter positive integers.*
- *No need to document your code. However, you may add comments if you think they are needed for clarity.*

Question 5 (30 points)

Implement the function:

```
int* intersectSortedArrays(int srtArr1[], int n1,
                           int srtArr2[], int n2,
                           int& resArrSize)
```

This function is given two base addresses of arrays of integers `srtArr1` and `srtArr2` and their logical sizes `n1` and `n2`. The elements in `srtArr1` and `srtArr2` are sorted, that is, they are ordered, in the arrays, in an ascending order.

When the function is called, it will create and return a new array, that contains all the elements that **appear in both arrays**. The function should also update the output parameter, `resArrSize`, with the logical size of the new array that was created.

For example:
if `srtArr1=[1, 3, 5, 6, 8, 14, 15]`,
and `srtArr2=[2, 3, 5, 10, 15, 18]`,
after calling:
`intersectSortedArrays(srtArr1, 7, srtArr2, 6, resArrSize)`,
the function should create and return an array that contains `[3, 5, 15]`, and update the value in `resArrSize` to be `3`.

Note: You may assume that each array does not contain duplicate items.

Implementation requirements:

Your function should run in **linear time**. That is, if n_1 and n_2 are the sizes of the input arrays, the runtime should be $\theta(n_1 + n_2)$.

Note: You don't need to write a `main()` program.

Question 6 (15 points)

Give a **recursive** implementation for the following function:

```
int minValueInEvenIndices(int arr[], int arrSize)
```

The function is given the base address of a non-empty array of integers `arr`, and its logical size `arrSize`.
When called, it should return the minimum value, out of all the elements **in even indices**. That is the minimum value out of `arr[0]`, `arr[2]`, `arr[4]`, ...

For example, if `arr=[7, 1, 5, 4, 2, 12, 15]`,
calling `minValueInEvenIndices(arr, 7)` should return `2`.
And if `arr=[7, 1, 5, 4, 2, 12]`,
calling `minValueInEvenIndices(arr, 6)` should return `2`.

Notes:

1. You don't need to write a `main()` program.
2. Your function **must be recursive**.