NYU, Tandon School of Engineering
Extended Bridge to CS — Spring 2024

Exam 2

Part 1: Theoretical

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.

Part 1: Mathematical Induction

Question 1

Use mathematical induction to prove that for all integers $n \geq 0$,

$$1 + 3 + 3^2 + 3^3 + \cdots + 3^n = \frac{3^{n+1} - 1}{2}.$$

Part 1: Introduction to Counting

Question 2

A class consists of **8** sophomores and **6** freshmen. The class needs to form a committee of size seven (**7**). How many committees are possible if the committee must have four (**4**) sophomores and three (**3**) freshmen? **Explain your answer.**

Question 3

In this question, consider all bit strings of length ten (**10**). How many bit strings of length **10** have equal numbers of **0**'s and **1**'s? **Explain your answer.**

Part 1: Discrete Probability

Question 4

A bowl has eight (**8**) ping pong balls numbered **1, 2, 2, 2, 3, 4, 5, 5**. You pick a ball at random. What is the probability that the number on the ball drawn is greater than or equal to **3**? **Explain your answer.**

Question 5

Suppose you flip a biased coin, where the probability of getting heads is $\frac{2}{3}$ and the probability of getting tails is $\frac{1}{3}$, eight (**8**) times. What is the probability of getting **at most 1 head** out of these **8 flips**? **Explain your answer.**

Part 1: Asymptotic Analysis of Algorithms

Question 6

Analyze its running time of `function1`. **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 function1(int n) {
    int i, j;
    int sum = 0;

    for (i = 1; i <= n; i += 3) {
        for (j = 1; j <= n; j++) {
            sum += (i + j);
        }
    }

    if (n % 2 == 0) {
        for (i = 1; i <= n; i++) {
            for (j = 1; j <= n; j++) {
                sum += (i + j);
            }
        }
    }

    return sum;
}
```

Question 7

Analyze its running time of `function2`. **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 function2(int n){
    int i, j, k;
    int sum = 0;

    for (k = 1; k <= n; k += 1) {
        for (i = 1; i <= n; i *= 2) {
            j = i;
            while (j >= 1) {
                sum += 1;
                j /= 2;
            }
        }
    }

    return sum;
}
```

Part 2: Coding Question 1

Question 8

Give a **recursive** C++ implementation for the function:

```
int countXs(int *arr, int n, int x)
```

The above function is given `arr`, a base address of an array that will contain **non-zero integers**, its logical size `n`, and another **non-zero** integer `x`. When this `countXs` function is called, it should return **the number of times `x` has appeared in `arr`.**

1. For example, if `arr` = {`1, -5, 2, -5, -5, 9, -5, 3`}, after calling `countXs(arr, 10, -5)`, this function should return **4** because -5 has appeared 4 times in `arr`.
2. For example, if `arr` = {`1, 5, 20, 12, 2, 5, -5, 7`}, after calling `countXs(arr, 8, 3)`, this function should return **0**. Because 3 has appeared 0 time in `arr`.
3. For example, if `arr` = {`-1, 3, 9, 3, 2, 3, -5, 3, 2, 3, -5`}, after calling `countXs(arr, 11, 3)`, this function should return **5** because 3 has appeared 5 times in `arr`.

Implementation requirements:

- Your function should run in **worst case Linear time**. That is, it should run in $\theta(n)$ where n = logical size of the array **arr**.
- Your function **must be recursive**.
- You are not allowed to use C++ syntactic features that were not covered in the Bridge program so far.
- Assume that zero (**0**) is not in the elements in *arr*.

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

Part 2: Coding Question 2

Question 9

Give a C++ implementation for the function:

```
void combineSortedArrays(int S1[], int S2[], int n1, int n2, vector<int> &CombinedArray);
```

The above function is given two sorted arrays `S1` and `S2` that will contain **integers in non-decreasing order**, an integer `n1` that will indicate the **logical size** of the array `S1`, an integer `n2` that will indicate the **logical size** of the array `S2`, and an address to an integer vector `CombinedArray` (type `vector<int>`) that will contain the merged integers **sorted in non-decreasing order**. When this `combineSortedArrays` function is called, it should merge the elements of `S1` and `S2` in sorted order in `CombinedArray` vector (type `vector<int>`) and remove duplicates. Note that `S1` will contain unique integers sorted in **non-decreasing order** and `S2` will contain unique integers sorted in **non-decreasing order**, but same elements may appear in both `S1` and `S2`. After merging, all the elements in `CombinedArray` vector (type `vector<int>`) should be **unique** (there will be no duplicate elements) and will be sorted in **non-decreasing order**.

1. For example, if `S1 = {-100, -75, -20, 15, 20}`, `S2 = {-75, -20, -10, 20}` and type of `CombinedArray` variable is `vector<int>`, after calling `combineSortedArrays(S1, S2, 4, 4, CombinedArray)`, `CombinedArray` should be {-100, -75, -20, -10, 15, 20}.
2. For example, if `S1 = {-1, 5, 15, 17, 26, 44}`, `S2 = {-10, 5, 24, 30, 50}` and type of `CombinedArray` variable is `vector<int>`, after calling `combineSortedArrays(S1, S2, 6, 5, CombinedArray)`, `CombinedArray` should be {-10, -1, 5, 15, 17, 24, 26, 30, 44, 50}.
3. For example, if `S1 = {-10, 5, 15}`, `S2 = {0, 15, 24, 30, 90}` and type of `CombinedArray` variable is `vector<int>`, after calling `combineSortedArrays(S1, S2, 3, 5, CombinedArray)`, `CombinedArray` should be {-10, 0, 5, 15, 24, 30, 90}.

Implementation requirements:

- Your function should run in $\theta(n1 + n2)$ where $n1$ = logical size of the array `S1` and $n2$ = logical size of the array `S2`. For simplicity, you can assume that amortized $\theta(1)$ is the same as $\theta(1)$.
- You are not allowed to use C++ syntactic features that were not covered in the Bridge program so far.
- You can assume that initial value of `CombinedArray` variable (type `vector<int>`) is Empty.

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