Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.

Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed**, we cannot accept hand-written solutions. Here's a short intro to **Latex**.
- In this homework we denote the asymptomatic *Big-O* notation by $\mathcal{O}$ and *Small-O* notation is represented as o.
- We recommend using online Latex editor **Overleaf**. Download the **.tex** file from Canvas and upload it on overleaf to edit.
- You should submit your work through **Gradescope** only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Canvas if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.

Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

Piazza threads for hints and further discussion

Piazza Threads
Question 1
Question 2
Question 3
Question 4

Recommended reading:
Greedy Algorithms: Ch. 16 16.1, 16.2, 16.3; Ch. 2 2.1, 2.2

Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

1. *(5 pts) Assume you run your Huffman tree algorithm and you produce the following pre-fix codes. Describe why there must be an error in your algorithm.*

```
S = 11
c = 10
i = 110
e = 100
n = 010
```

There is an error in the algorithm since S and c are considered non-terminals in the tree and not leaf nodes. Due to this, it is not obvious whether or not it should be treated as a c or an e.

Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

2. *(5 × 2 = 10 pts) Consider the given Huffman Tree.*

huffman.png

(a) *Decode the string* ***"1001111101001000110110100 1"*** *encoded using the above Huffman encoding tree.*

The decoded string is "SKOBUFFS". Shown below
"S" = 1001
"K" = 111
"O" = 101
"B" = 00
"U" = 1000
"F" = 110
"F" = 110
"S" = 1001

(b) Decode the string ***"1011110010001001"*** *encoded using the above Huffman encoding tree.*

The decoded string is "OKBUS". Shown below
"O" = 101
"K" = 111
"B" = 00
"U" = 1000
"S" = 1001

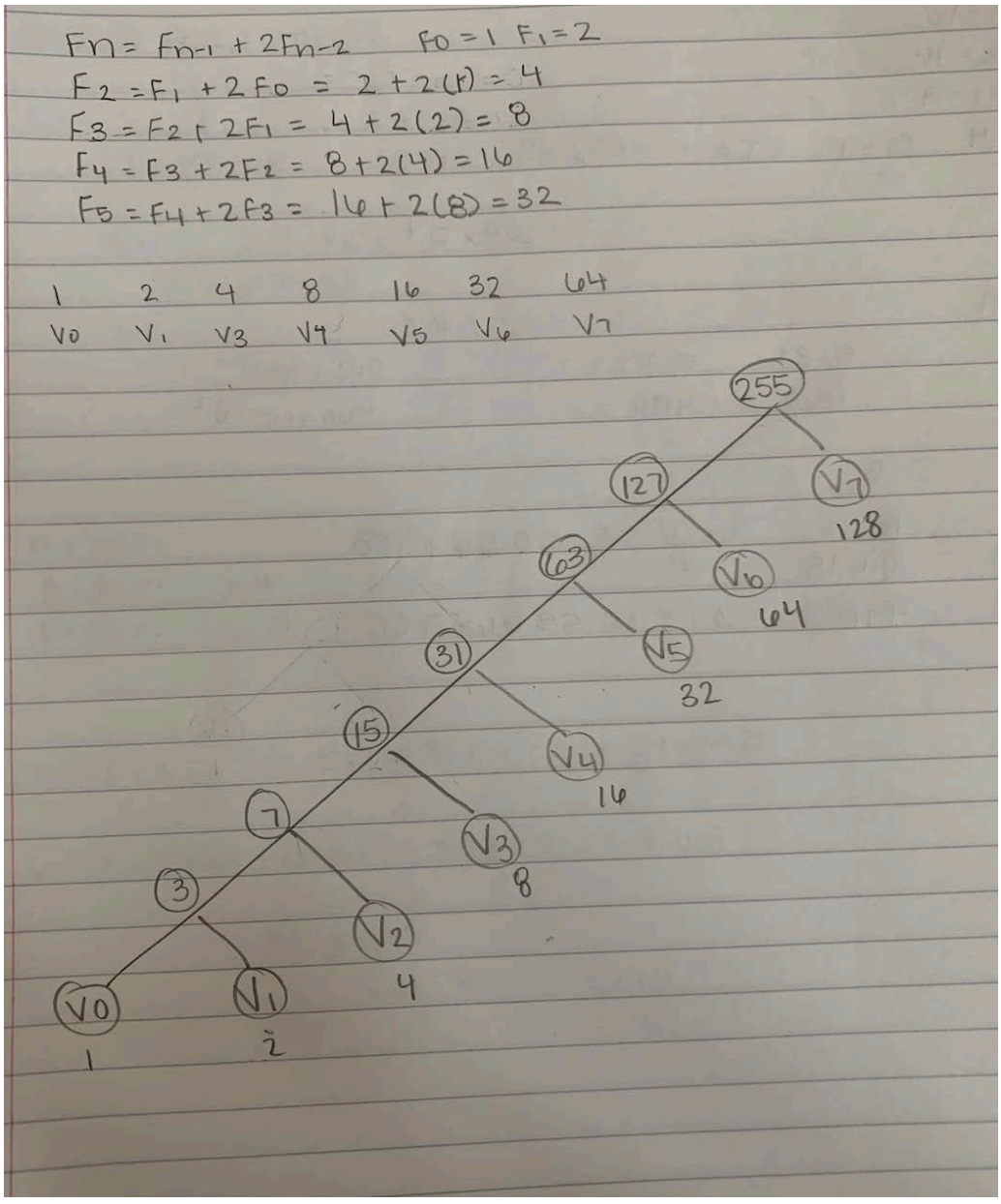
Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 3A (55 points) **Summer 2020, CU-Boulder**

3. *(15 pts) Consider the recurrence* $F_n = F_{n-1} + 2F_{n-2}$*, with the base cases* $F_0 = 1$ *and* $F_1 = 2$*. Suppose we have letters* $v_0, \ldots, v_7$*; the frequency of* v_i *is given by* F_i*, where* $i \in \{0, 1, 2...6, 7\}$*. Draw a Huffman tree for* $v_0, \ldots, v_7$*.*



Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

4. *(25 pts) In this question we consider the changemaking problem, of making change for n cents using the smallest number of coins. Suppose we have coins with denominations of $v_1 > v_2 > ...v_r$ for r coins types, where each coin's value v_i is a positive integer. Your goal is to determine how many coins of each denomination d_i are required to reach the sum n, such that the total number of coins used is minimized (i.e the value of $\sum_{i=1}^{r} d_i = k$ is minimized).*

   Note that the sum of all included coins should be n. (i.e $\sum_{i=1}^{r} d_i v_i = n$)

   (a) *(15 pts) A greedy algorithm for making change is the **cashier's algorithm**. In cashier's algorithm at each iteration we add a coin of highest value ensuring that it does not take us past the value n. The above step is repeated until the sum reaches n. If it is not possible to reach the sum n the algorithm returns no solution as the answer. Consider the following pseudocode meant to implement the cashier's algorithm where n is the amount of money to make change for and v is a vector of the coin denominations sorted in descending order and the vector d should hold the count of number of coins in each denomination.*

```
get_change(n, v, r):
{
    d[1...r] = 1
    while(n>0)
    {
        k=r
        while(k>0 and v[k]>n)
        {
            k++
        }
        if(k <= 0)
        {
            return 'no solution'
        }
        else
        {
            n = n-v[k]
        }
    }
    return d
}
```

Name: Pourna Sengupta

ID: 109086577

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 3A (55 points) **Summer 2020, CU-Boulder**

This code has bugs. Identify the bugs and explain why each would cause the algorithm to fail.

(b) (10 *pts*) *Assume that a country has the following coin denominations.* $\{\$1, \$6, \$10, \$15\}$. *Provide two examples for which the greedy algorithm does not yield an optimal solution for making change. In both the examples also show the optimal solution adds that adds up to the same value using smaller set of coins.*

Name: Pourna Sengupta
ID: 109086577

5. ***Extra Credit (5% of total homework grade)*** *For this extra credit question, please refer the leetcode link provided below or click here. Multiple solutions exist to this question ranging from brute force to the most optimal one. Points will be provided based on Time and Space Complexities relative to that of the most optimal solution.*

   Please provide your solution with proper comments which carries points as well.

   https://leetcode.com/problems/jump-game/

   ```
   Replace this text with your source code inside of the .tex document
   ```