# CS3230 PA2

**Yadunand Prem, A0253252M**

## Task 1

### A

No I did not need to implement Strassen's Algorithm. The Freivalds algorithm allows us to calculate teh solution of $A \times B = C$ in $O(kn^2)$ time, where k is a chosen constant, which is faster than Strassen's Algorithm. The benefits that strassens algorithm would give us is the 100% accuracy, but with a large enough $k$ value, Freivalds Algorithm is good enough.

### B

Monte Carlo

### C

#### Run faster than worst-case time complexity?

TC2, as the frievald's algorithm terminates early when the multiplication is wrong. For example, if $k = 8$,but in the first iteration, $A \times Br \neq Cr$, then this terminates and returns "WA".

#### ii Always correct vs may be wrong?

TC1 is always correct. Since the matrix multiplication is correct, any operation done on any given row, $A \times Br = Cr$ will always be correct, and thus, the algorithm will return "AC".

TC2 operations on the other hand, may return the wrong result. Since the columns are chosen at random, there could be the case where $A \times B \neq C$, but the chosen $r$ gives $A \times Br = Cr$. Since the algorithm doesn't check every column on every row, the result might be wrong.

### D

I chose $k == 8$ cause after trying with a few $k$ values, 8 was what had passed the testcases. When I had initially picked 6, $1 - \frac{1}{2^6} \approx 0.98$, it didn't work, but with 8, $1 - \frac{1}{2^8} \approx 0.99$, it worked

### E

No of tries = $\frac{1}{\left(1-\left(\frac{1}{2^8}\right)\right)^{500}} \approx 8$, just looking at the 500 type 2 operations from the above given TC2.

## Task 2

### A

The subproblem can be represented as `DP(i, at_S)`, where `at_S` is the side of the road Cissi is on, and it returns the number of crossings needed from Home to `i`.

Thus, at the beginning, `DP(0, 0) = 0`, and `DP(0, 1) = 1`, as to get to the south side, she needs to cross once

- Base Case
  - `DP(0, 0) = 0` as She is at Home
  - `DP(0, 1) = 1` as she needs to cross the road once to reach the south side
- General Case
  - If `crossing[i] = 'B'`
    - `DP(i, 0) = DP(i-1, 0) + 1`
    - `DP(i, 1) = DP(i-1, 1) + 1`

    - Basically, +1 on both north and south side
  - If `crossing[i] = 'N'`
    - `DP(i,0) = min(DP(i-1,0) + 1, DP(i-1,1) + 1)`
      - to get to the North side, its either you cross the NORTH street from the north side of the road OR you cross the horizontal street from the south side of the road
    - `DP(i,1) = min(DP(i-1,0) + 1 + 1, DP(i-1,1))`
      - Basically, to get to the south side, its either you cross the NORTH street from the north side of the road and cross the horizontal street OR you don't cross at all from teh south side of the street
  - If `crossing[i] = 'S'`
    - `DP(i,0) = min(DP(i-1,0), DP(i-1,1) + 2)`
    - `DP(i,1) = min(DP(i-1,0) + 1, DP(i-1,1)) + 1`

The final result will be in `DP(n-1, 0)` as the school is on the north side of the road at `n-1`

## B

- Base Case
  - `DP[i][0] = 0`
  - `DP[i][1] = 1`

## C

$O(2^n)$

## D

$O(n)$