Amazon

300 marks test, 1.5 hr on hackerearth (2 coding questions - 200 marks, 20 MCQ - 100 marks)
https://www.geeksforgeeks.org/minimum-number-platforms-required-railwaybus-station/ (Same question - 100 marks)
https://www.geeksforgeeks.org/count-integral-points-inside-a-triangle/ (For a polygon instead of a triangle - 100 marks)
20 MCQ (5 marks each) - OOPS(4-5), Networks(5-6), OS(3-4), Data Structures (5), Algorithms (2-3)
Running code https://ideone.com/laser0/amazon

Walmart

1. Given 3 unsorted arrays A, B and C you need to find all possible combinations such that A[i] + B[j] = B[k] + C[l].

Pseudo Code:

```
for (a in A) {
  for (b in B) {
     insert(a+b, hashtable);
  }
}

for (b in B) {
   for (c in C) {
      if (b+c in hashtable) {
         // blah
      }
   }
}
```

2. A string is beautiful if the string `S = T + T`. Given a string, find longest beautiful subsequence.

It was an online test of 90 minutes and was conducted on Hackerearth. It consisted of 10-12 MCQ’s and 3 coding questions. MCQ’s consisted of general aptitude questions, questions related to networking, programming, C input/output etc .

One coding question was – given arrival and departure time of guests in a party … find no of plates or something .Similar to this
https://www.geeksforgeeks.org/minimum-number-platforms-required-railwaybus-station/

Unacademy

What are Unicasting, Anycasting, Multicasting and Broadcasting?

- If the message is sent from a source to a single destination node, it is called **Unicasting**. This is typically done in networks.
- If the message is sent from a source to a any of the given destination nodes. This is used a lot in Content delivery Systems where we want to get content from any server.
- If the message is sent to some subset of other nodes, it is called **Multicasting**. Used in situation when there are multiple receivers of same data. Like video conferencing, updating something on CDN servers which have replica of same data.
- If the message is sent to all the nodes in a network it is called **Broadcasting**. This is typically used in Local networks, for examples DHCP and ARP use broadcasting.

What are layers in OSI model?

There are total 7 layers
1. Physical Layer
2. Data Link Layer
3. Network Layer
4. Transport Layer
5. Session Layer
6. Presentation Layer
7. Application Layer

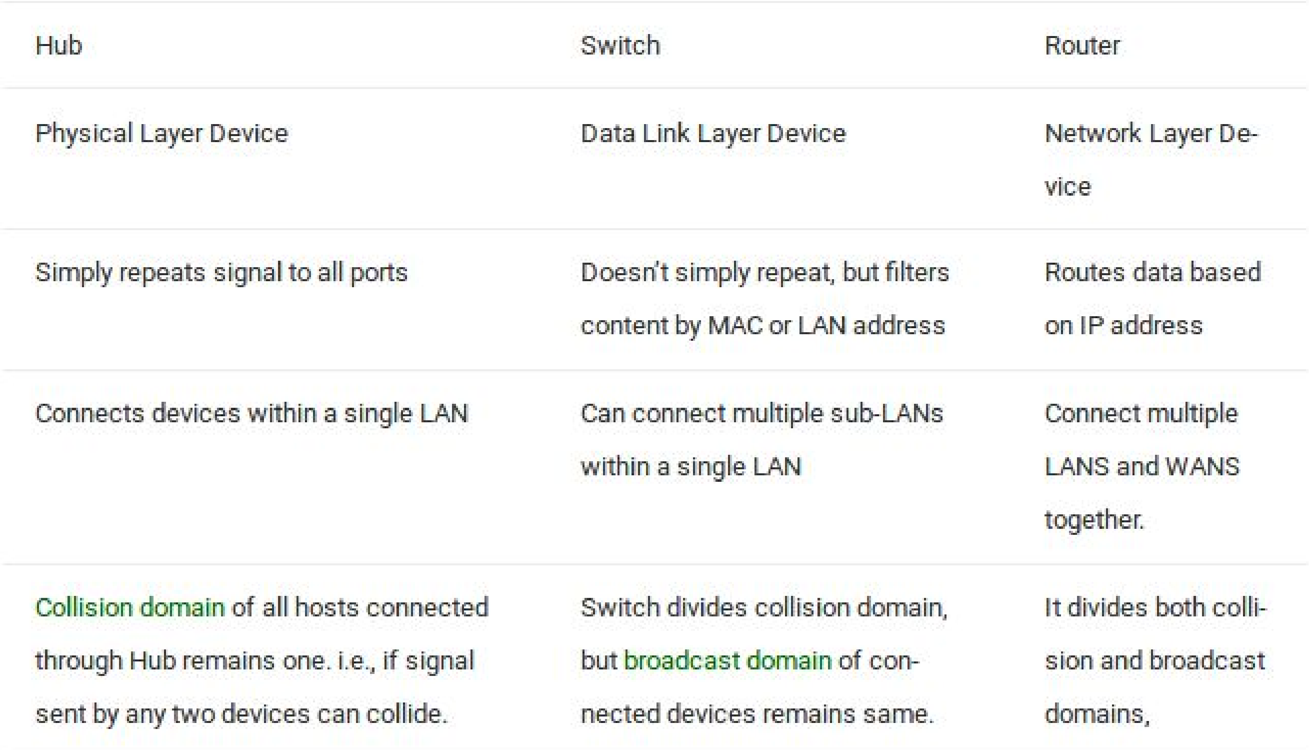
What is Stop-and-Wait Protocol?

In Stop and wait protocol, a sender after sending a frame waits for acknowledgement of the frame and sends the next frame only when acknowledgement of the frame has received.

What is Piggybacking?

Piggybacking is used in bi-directional data transmission in the network layer (OSI model). The idea is to improve efficiency piggy back acknowledgement (of the received data) on the data frame (to be sent) instead of sending a separate frame.

Differences between Hub, Switch and Router?



Hub	Switch	Router
Physical Layer Device	Data Link Layer Device	Network Layer Device
Simply repeats signal to all ports	Doesn't simply repeat, but filters content by MAC or LAN address	Routes data based on IP address
Connects devices within a single LAN	Can connect multiple sub-LANs within a single LAN	Connect multiple LANS and WANS together.
Collision domain of all hosts connected through Hub remains one. i.e., if signal sent by any two devices can collide.	Switch divides collision domain, but broadcast domain of connected devices remains same.	It divides both collision and broadcast domains,

What happens when you type a URL in web browser?

A URL may contain request to HTML, image file or any other type.

1. If content of the typed URL is in cache and fresh, then display the content.
2. Else find IP address for the domain so that a TCP connection can be setup. Browser does a DNS lookup.
3. Browser needs to know IP address for a url, so that it can setup a TCP connection. This is why browser needs DNS service. Browser first looks for URL-IP mapping browser cache, then in OS cache. If all caches are empty, then it makes a recursive query to the local DNS server. The local DNS server provides the IP address.
4. Browser sets up a TCP connection using three way handshake.
5. Browser sends a HTTP request.
6. Server has a web server like Apache, IIS running that handles incoming HTTP request and sends a HTTP response.
7. Browser receives the HTTP response and renders the content.

What is DHCP, how does it work?

1. The idea of DHCP (Dynamic Host Configuration Protocol) is to enable devices to get IP address without any manual configuration.
2. The device sends a broadcast message saying “I am new here”
3. The DHCP server sees the message and responds back to the device and typically allocates an IP address. All other devices on network ignore the message of new device as they are not DHCP server.

In Wi Fi networks, Access Points generally work as a DHCP server.

What is ARP, how does it work?
ARP stands for Address Resolution Protocol. ARP is used to find LAN address from Network address. A node typically has destination IP to send a packet, the nodes needs link layer address to send a frame over local link. The ARP protocol helps here.

1. The node sends a broadcast message to all nodes saying what is the MAC address of this IP address.
2. Node with the provided IP address replies with the MAC address.

Like DHCP, ARP is a discovery protocol, but unlike DHCP there is not server here.

Socket:
The unique combination of IP address and Port number together are termed as Socket.

DNS Server:
DNS stands for **Domain Name system**.
DNS is basically a server which translate web addresses or URL (ex: www.google.com) into their corresponding IP addresses. We don’t have to remember all the IP addresses of each and every website.
The command ‘**nslookup**’ gives you the IP address of the domain you are looking for.

Port:
Port can be referred as logical channel through which data can be sent/received to an application.

IP Address (Internet Protocol address):
Also know as Logical Address, is the network address of the system across the network.

ARP:

ARP stands for **Address Resolution Protocol**.
It is used to convert the IP address to its corresponding Physical Address(i.e.MAC Address).
ARP is used by the Data Link Layer to identify the MAC address of the Receiver’s machine.

RARP:
RARP stands for **Reverse Address Resolution Protocol**.
As the name suggest, it provides the IP address of the a device given physical address as input.
But RARP has become obsolete since the time DHCP has come into picture.

transport layer protocols is used to support electronic mail?
E-mail uses SMTP as application layer protocol. SMTP uses TCP as transport layer protocol**.**

In the slow start phase of the TCP congestion control algorithm, the size of the congestion window increases exponentially.

Puzzles : https://www.geeksforgeeks.org/category/puzzles/

===
Average of a and b that does not cause overflow : (a / 2) + (b / 2) + ((a % 2 + b % 2) / 2)

How are virtual functions implemented at a deep level?

From "Virtual Functions in C++"
Whenever a program has a virtual function declared, a v - table is constructed for the class. The v-table consists of addresses to the virtual functions for classes that contain one or more virtual functions. The object of the class containing the virtual function contains a virtual pointer that points to the base address of the virtual table in memory. Whenever there is a virtual function call, the v-table is used to resolve to the function address. An object of the class that contains one or more virtual functions contains a virtual pointer called the vptr at the very beginning of the object in the memory. Hence the size of the object in this case increases by the size of the pointer. This vptr contains the base address of the virtual table in memory. Note that virtual tables are class specific, i.e., there is only one virtual table for a class irrespective of the number of virtual functions it contains. This virtual table in turn contains the base addresses of one or more virtual functions of the class. At the time when a virtual function is called on an object, the vptr of that object provides the base address of the virtual table for that class in memory. This table is used to resolve the function call as it contains the addresses of all the virtual functions of that class. This is how dynamic binding is resolved during a virtual function call.

===

Nvidia

- What is IPC and how semaphore is used for this?
    - Inter Process Communication
    - multiple processes are using the same region of code and if all want to access parallelly then the outcome is overlapped.
    - if all the jobs start parallelly, then one user output is overlapped with another. So, we need to protect that using semaphores i.e., locking the critical section when one process is running and unlocking when it is done. This would be repeated for each user/process so that one job is not overlapped with another job.
    - Basically semaphores are classified into two types −
        - **Binary Semaphores** − Only two states 0 & 1, i.e., locked/unlocked or available/unavailable, Mutex implementation.
        - **Counting Semaphores** − Semaphores which allow arbitrary resource count are called counting semaphores.
    - To perform synchronization using semaphores, following are the steps −

Step 1 − Create a semaphore or connect to an already existing semaphore (semget())

Step 2 − Perform operations on the semaphore i.e., allocate or release or wait for the resources (semop())

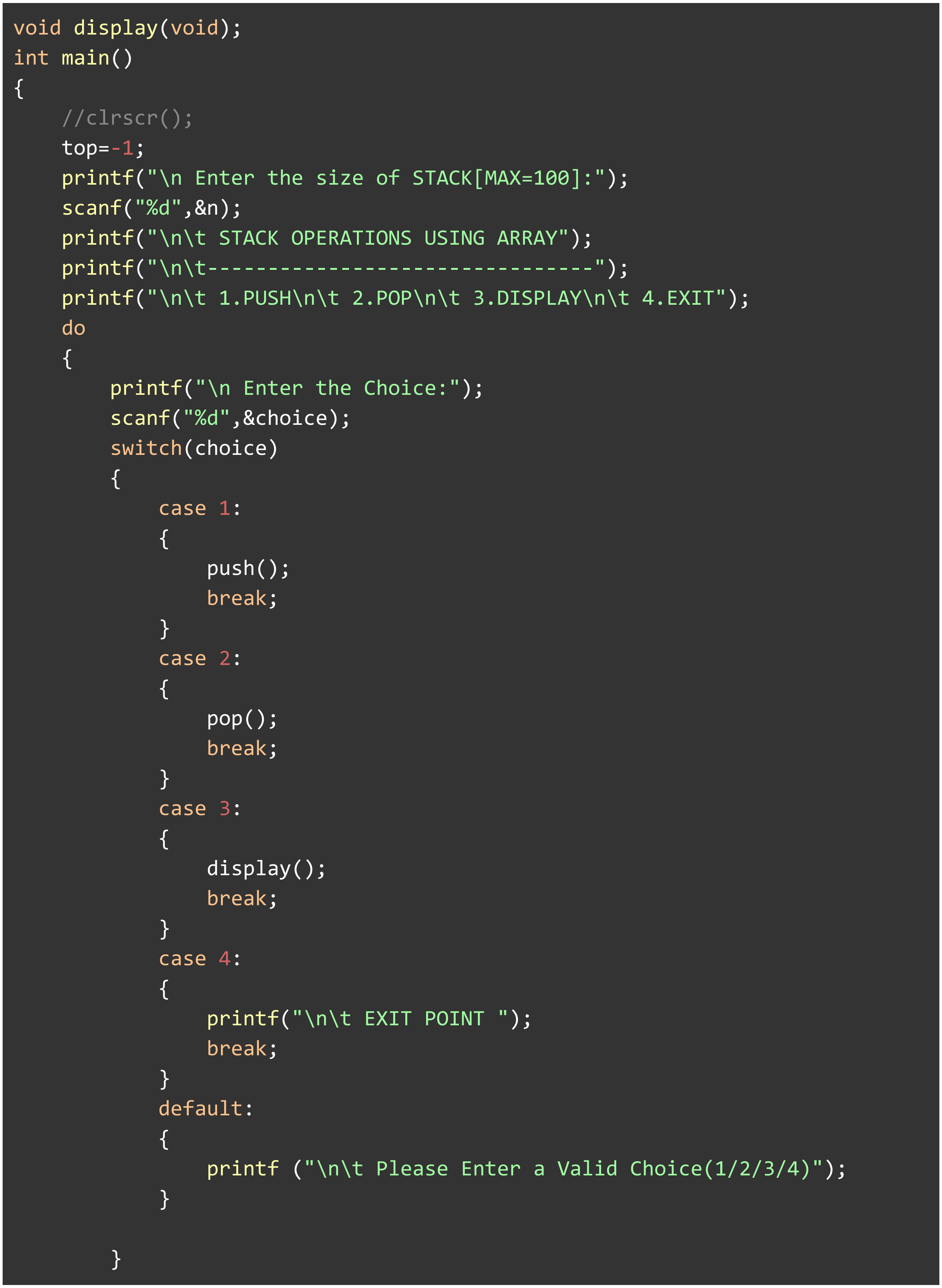
Step 3 − Perform control operations on the message queue (semctl())

- What’s wrong in the given code?

```
char* func(){
char name[] = “name”;
return name;
}


int main(){
char* ptr = func();
printf(“%s”,&ptr);
}
```

- Implement stack in C?

```
#include<stdio.h>

int stack[100],choice,n,top,x,i;
void push(void);
void pop(void);
```



```c
void display(void);
int main()
{
    //clrscr();
    top=-1;
    printf("\n Enter the size of STACK[MAX=100]:");
    scanf("%d",&n);
    printf("\n\t STACK OPERATIONS USING ARRAY");
    printf("\n\t--------------------------------");
    printf("\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT");
    do
    {
        printf("\n Enter the Choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            {
                push();
                break;
            }
            case 2:
            {
                pop();
                break;
            }
            case 3:
            {
                display();
                break;
            }
            case 4:
            {
                printf("\n\t EXIT POINT ");
                break;
            }
            default:
            {
                printf ("\n\t Please Enter a Valid Choice(1/2/3/4)");
            }

        }
    }
    while(choice!=4);
    return 0;
}
void push()
{
    if(top>=n-1)
    {
        printf("\n\tSTACK is over flow");

    }
    else
    {
        printf(" Enter a value to be pushed:");
        scanf("%d",&x);
        top++;
        stack[top]=x;
    }
}
void pop()
{
    if(top<=-1)
    {
        printf("\n\t Stack is under flow");
    }
    else
    {
        printf("\n\t The popped elements is %d",stack[top]);
        top--;
    }
}
```

- what is the difference between C and C++?
    - C supports procedural programming paradigm for code development. C++ supports both procedural and object oriented programming paradigms; therefore C++ is also called a hybrid language.
    - C does not support object oriented programming; therefore it has no support for polymorphism, encapsulation, and inheritance.
      Being an object oriented programming language C++ supports polymorphism, encapsulation, and inheritance.

    - C does not support function and operator overloading.C++ supports both function and operator overloading.
    - C does not allow functions to be defined inside structures. In C++, functions can be used inside a structure.
    - C provides malloc() and calloc() functions for dynamic memory allocation, and free() for memory de-allocation. C++ provides new operator for memory allocation and delete operator for memory de-allocation.

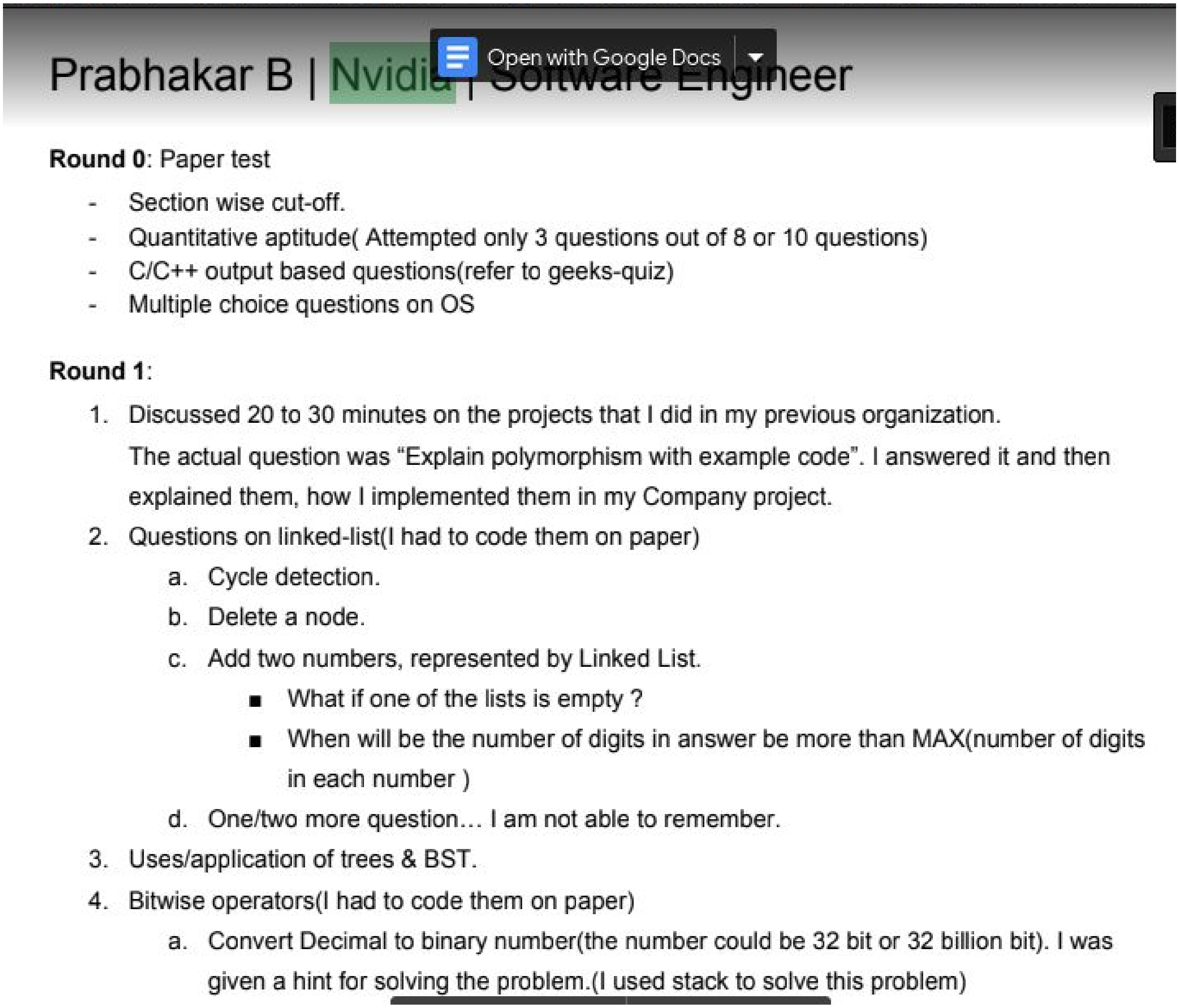
- What is the difference between stacks or queues?

  Stack is a collection of objects that works in LIFO (Last in First out) mechanism while Queue is FIFO (First in First out). This means that the object that is inserted first is removed last in a stack while an object that is inserted first is removed first in a queue.

  Stack – Represents the collection of elements in Last In First Out order. Operations includes testing null stack, finding the top element in the stack, removal of top most element and adding elements on the top of the stack.

  Queue - Represents the collection of elements in First In First Out order. Operations include testing null queue, finding the next element, removal of elements and inserting the elements from the queue. Insertion of elements is at the end of the queue Deletion of elements is from the beginning of the queue.

- Describe your project and what are its applications?
- Describe how to implement matrix multiplication in CUDA? (If mentioned in resume)
- What is inheritance and how to implement it in C?
- What is polymorphism give an example?
- What is perceptron and how it is different from neural network?
- What is deep learning or deep network?
- How to implement unsupervised learning in deep neural networks?
- what is learned in deep learning?



Prabhakar B | Nvidia | Software Engineer

Round 0: Paper test

- Section wise cut-off.
- Quantitative aptitude(Attempted only 3 questions out of 8 or 10 questions)
- C/C++ output based questions(refer to geeks-quiz)
- Multiple choice questions on OS

Round 1:

1. Discussed 20 to 30 minutes on the projects that I did in my previous organization.
   The actual question was “Explain polymorphism with example code”. I answered it and then explained them, how I implemented them in my Company project.
2. Questions on linked-list(I had to code them on paper)
   a. Cycle detection.
   b. Delete a node.
   c. Add two numbers, represented by Linked List.
      - What if one of the lists is empty ?
      - When will be the number of digits in answer be more than MAX(number of digits in each number)
   d. One/two more question… I am not able to remember.
3. Uses/application of trees & BST.
4. Bitwise operators(I had to code them on paper)
   a. Convert Decimal to binary number(the number could be 32 bit or 32 billion bit). I was given a hint for solving the problem.(I used stack to solve this problem)

https://www.geeksforgeeks.org/rotate-matrix-180-degree/

https://www.geeksforgeeks.org/nvidia-interview-experience-set-4-on-campus/
https://www.geeksforgeeks.org/boggle-set-2-using-trie/
https://www.geeksforgeeks.org/nvidia-interview-set-3-campus/
https://www.geeksforgeeks.org/nvidia-interview-set-2-on-campus-for-r-d-team-pune/
https://www.geeksforgeeks.org/write-an-efficient-c-program-to-reverse-bits-of-a-number/
https://www.geeksforgeeks.org/compute-average-two-numbers-without-overflow/

Interview:
https://www.geeksforgeeks.org/nvidia-interview-experience-set-5/

Round 1-

Sub: - Computer organization

Q] Draw Von Neumann architecture.Explain its components. Q] Need for DMA(Direct Memory Access) controller? Explain in detail. Sub: - Operating System

Q] Internals of OS.

Q] Requirement for implementation of Multiprogrammed OS.

Q] System calls and Shift from User Mode to Kernel Mode.

Q] Semaphore(counting and binary).

Q] Producer Consumer Program with Semaphores. Q] Deadlock prevention Strategies. Sub:- Programming

Q] Find second Max from list of an array O(n) solution. Q] Find nth Fibonacci number Exponential ,O(n) and O(log n) solution. And Project Discussion

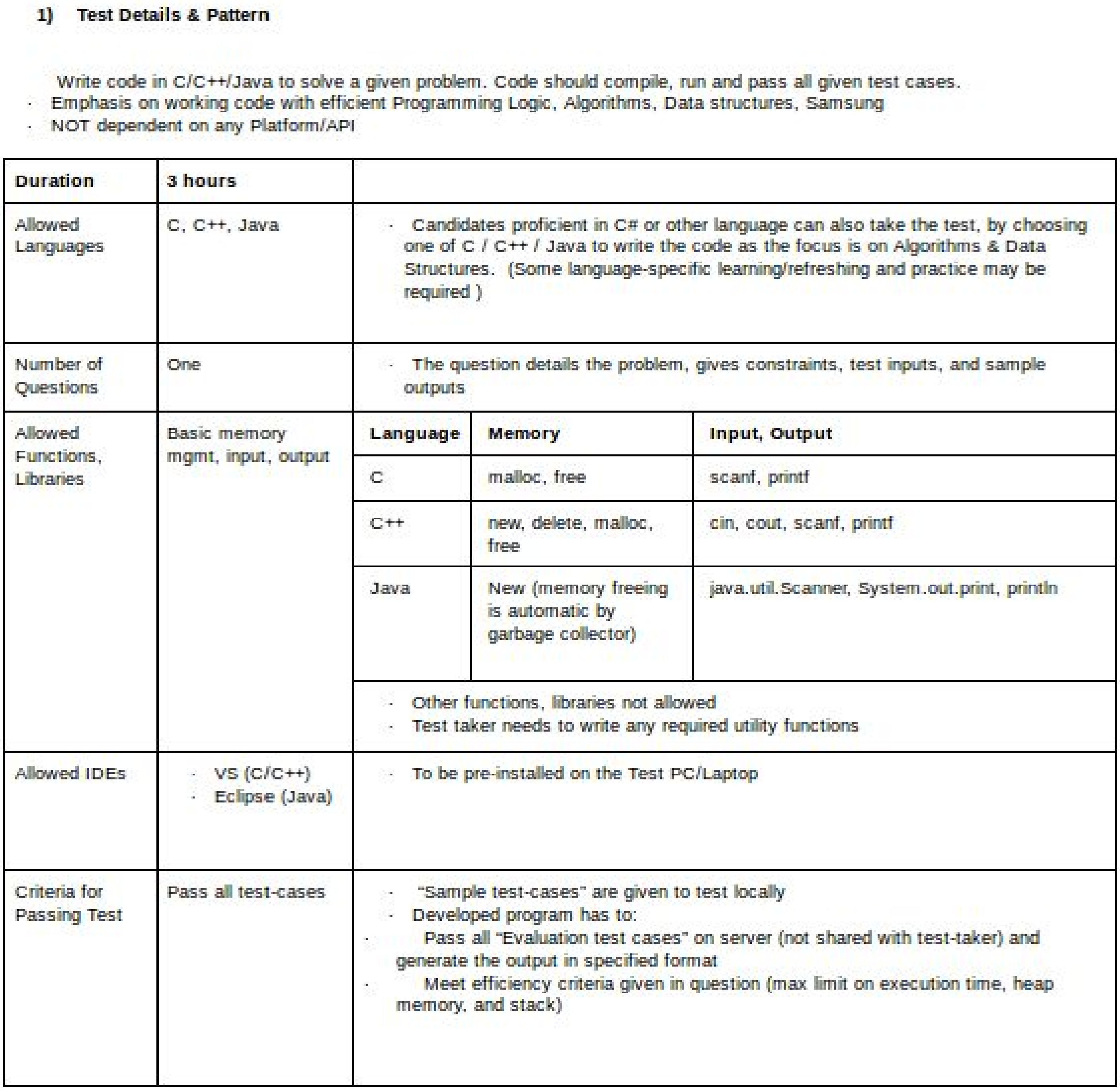
Round 2:-

Q] Already written code need to analyse what is does and time Complexity? And change the code to optimize it.

Q] Write code for finding the intersection point of two Linked lists. Q] One Probability based question Q] Given random diagram(Mixture of Chess and Mountain like structure) need to find number of Squares.

Q] Semaphore(wait and signal internal code)

Samsung



1) Test Details & Pattern

Write code in C/C++/Java to solve a given problem. Code should compile, run and pass all given test cases.

- Emphasis on working code with efficient Programming Logic, Algorithms, Data structures, Samsung
- NOT dependent on any Platform/API

<table>
<tr><td><b>Duration</b></td><td><b>3 hours</b></td><td colspan="3"></td></tr>
<tr><td>Allowed Languages</td><td>C, C++, Java</td><td colspan="3">· Candidates proficient in C# or other language can also take the test, by choosing one of C / C++ / Java to write the code as the focus is on Algorithms &amp; Data Structures. (Some language-specific learning/refreshing and practice may be required)</td></tr>
<tr><td>Number of Questions</td><td>One</td><td colspan="3">· The question details the problem, gives constraints, test inputs, and sample outputs</td></tr>
<tr><td rowspan="5">Allowed Functions, Libraries</td><td rowspan="5">Basic memory mgmt, input, output</td><td><b>Language</b></td><td><b>Memory</b></td><td><b>Input, Output</b></td></tr>
<tr><td>C</td><td>malloc, free</td><td>scanf, printf</td></tr>
<tr><td>C++</td><td>new, delete, malloc, free</td><td>cin, cout, scanf, printf</td></tr>
<tr><td>Java</td><td>New (memory freeing is automatic by garbage collector)</td><td>java.util.Scanner, System.out.print, println</td></tr>
<tr><td colspan="3">· Other functions, libraries not allowed
· Test taker needs to write any required utility functions</td></tr>
<tr><td>Allowed IDEs</td><td>· VS (C/C++)
· Eclipse (Java)</td><td colspan="3">· To be pre-installed on the Test PC/Laptop</td></tr>
<tr><td>Criteria for Passing Test</td><td>Pass all test-cases</td><td colspan="3">· "Sample test-cases" are given to test locally
· Developed program has to:
· Pass all "Evaluation test cases" on server (not shared with test-taker) and generate the output in specified format
· Meet efficiency criteria given in question (max limit on execution time, heap memory, and stack)</td></tr>
</table>

Can we use vector , stack , queue etc from #include<vector>, #include<stack> etc ? In test these worked. ??? (Nope you can't use any of the above asked libraries) what does it mean ??? Was string class allowed in cpp ?(nope you can't use string , only the functions mentioned in above table are allowed) was sort() function allowed? (you will have to implement sort function if required sort() not allowed)

1. Test was of 3 hour. Only 5 submission allowed(compile as many times you want). 10 test cases to be passed gscs.samsung.com/download/gscs103.zip] Cycle in directed graph number of vertex(n), number of edge(m). Then in next line m pairs of numbers representing edges of directed graph. Find if there is some cycle. If yes, print cycle in ascending order of vertex numbers involved in the cycle else print 0 (if there are multiple cycles print any one)Total no. of test cases: 10   None of them had the “No Cycle” graph.Samsung
2. 2-coloring in Undirected graph Given an undirected graph, if the graph can be coloured in two colors such that no two adjacent vertices are of same color then print the vertices which belong to the same color (you can print vertices with color 0/1), otherwise print -1.
   Input : First line gives number of vertices(V) and edges(E) (e.g. 7 10)
   Next line contains E pairs representing edges.
3. Test was of 3 hour. Only 10 submission allowed(compile as many times you want). 50 test cases to be passed
   Question: Constraints: N <= 50; M <= 50
   Inefficient Solution (passes all test cases): Consider jump of length = 1 to N-1 and for each case check if Destination is
   reachable. For first jump_length we reach destination return that.

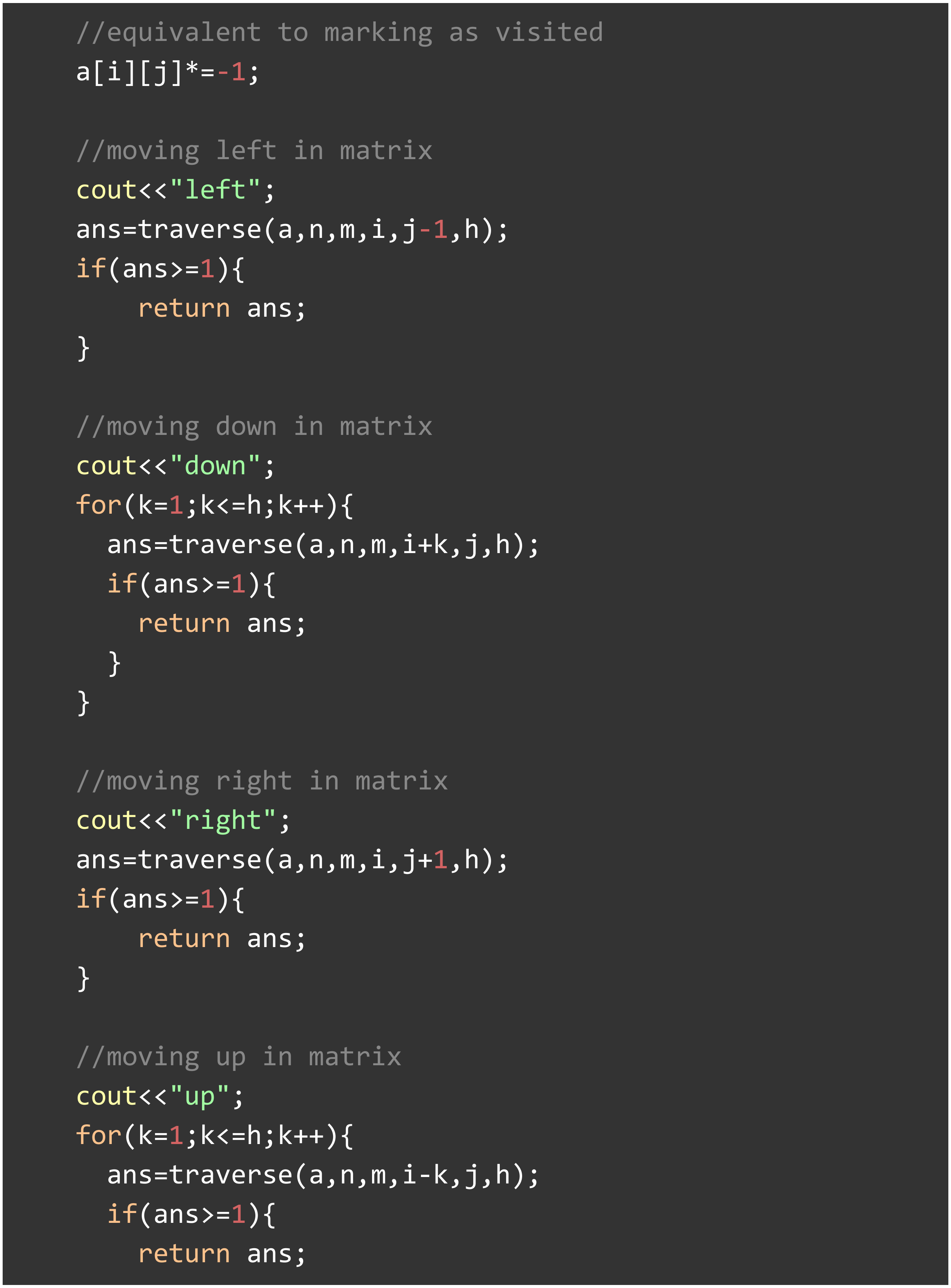
   ```
   Start at bottom {
   For jump_length in 1 to N-1 {
   For all continuous 1’s in that row{e
           For (current row - max_length) to (current row + max_length) {
                   Visit node if not visited;
   ```

QUESTION:

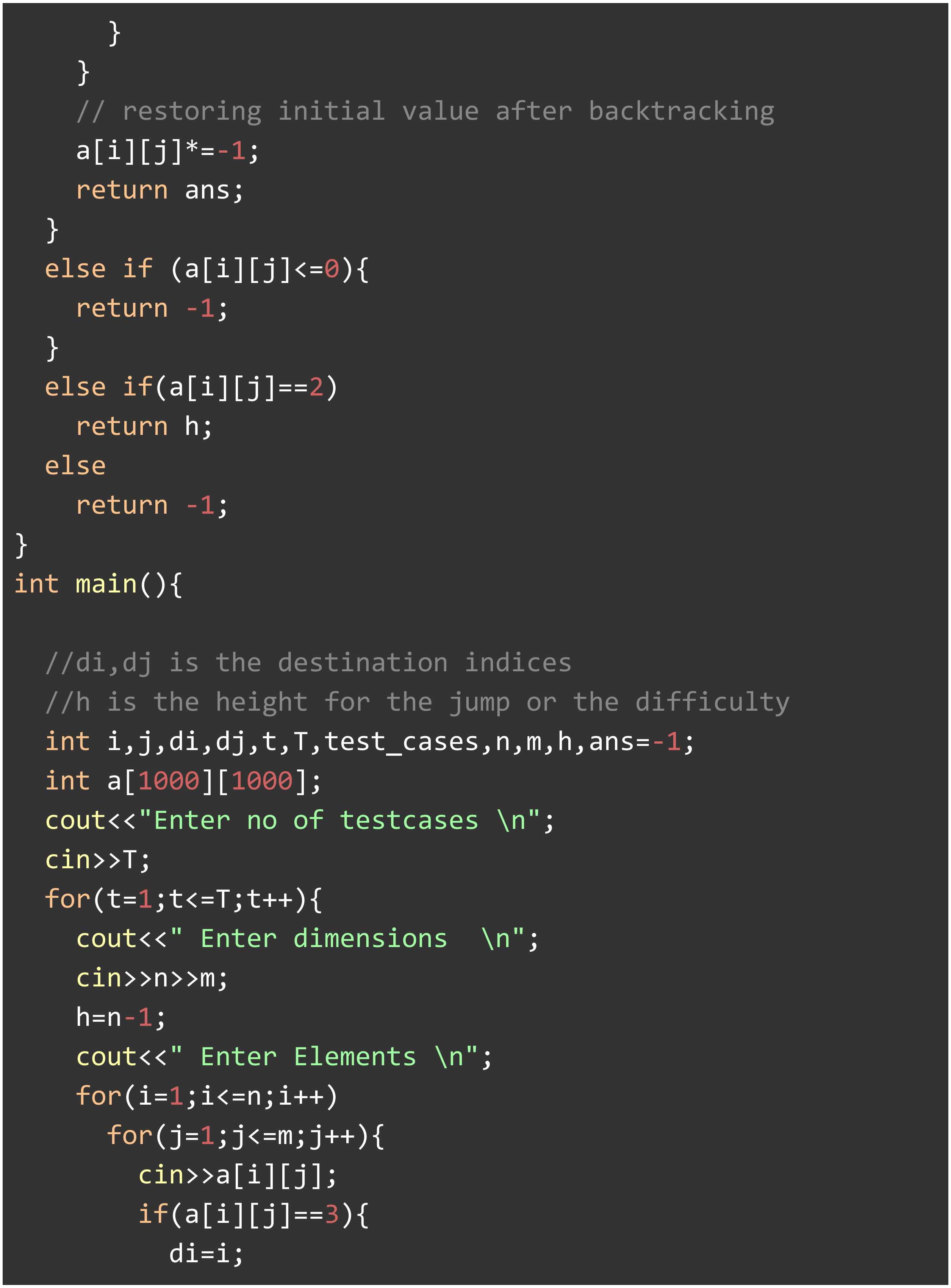
FIND THE DIFFICULTY OF THE CLIMB.

```
Given a grid( nXm matrix) containing values 0,1,2 and 3.
0 represents no path.
```

```
1 represent an existing path.
2 is the starting point.
3 is the destination.
Starting point is always at left bottom matrix[n,1].
Destination can be anywhere in the matrix. It is assured
that a path exist.
A rockclimber can move right or left if the adjacent
element is also 1.
The rockclimber however climb up or down skip any number of
rows the more rows he skip the greater will the difficulty.
*/
#include<iostream>
using namespace std;
void printMat(int a[1000][1000],int n,int m){
  int i,j;
  cout<<"\n\n";
  for(i=1;i<=n;i++){
    for(j=1;j<=m;j++){
      cout<<a[i][j]<<" ";
    }
    cout<<endl;
  }
}
int traverse(int a[1000][1000],int n,int m,int i,int j,int
h){
  int ans,k;
  printMat(a,n,m);
  //cout<<i<<" "<<j;
  if(i<=0 or j<=0 or i>n or j>m)
    return -1;
  if(a[i][j]==1 or a[i][j]==3){
    //for avoiding cycles during traversal,
```



```
    //equivalent to marking as visited
    a[i][j]*=-1;

    //moving left in matrix
    cout<<"left";
    ans=traverse(a,n,m,i,j-1,h);
    if(ans>=1){
        return ans;
    }

    //moving down in matrix
    cout<<"down";
    for(k=1;k<=h;k++){
      ans=traverse(a,n,m,i+k,j,h);
      if(ans>=1){
        return ans;
      }
    }

    //moving right in matrix
    cout<<"right";
    ans=traverse(a,n,m,i,j+1,h);
    if(ans>=1){
        return ans;
    }

    //moving up in matrix
    cout<<"up";
    for(k=1;k<=h;k++){
      ans=traverse(a,n,m,i-k,j,h);
      if(ans>=1){
        return ans;
```



```
      }
    }
    // restoring initial value after backtracking
    a[i][j]*=-1;
    return ans;
  }
  else if (a[i][j]<=0){
    return -1;
  }
  else if(a[i][j]==2)
    return h;
  else
    return -1;
}
int main(){

  //di,dj is the destination indices
  //h is the height for the jump or the difficulty
  int i,j,di,dj,t,T,test_cases,n,m,h,ans=-1;
  int a[1000][1000];
  cout<<"Enter no of testcases \n";
  cin>>T;
  for(t=1;t<=T;t++){
    cout<<" Enter dimensions  \n";
    cin>>n>>m;
    h=n-1;
    cout<<" Enter Elements \n";
    for(i=1;i<=n;i++)
      for(j=1;j<=m;j++){
        cin>>a[i][j];
        if(a[i][j]==3){
          di=i;
          dj=j;
        }
      }
    for(i=1;i<=h;i++){
        ans=traverse(a,n,m,di,dj,i);
        if(ans>=1)
          break;
    }
    cout<<"#"<<t<<" "<<ans<<endl;
  }
  return 0;
}
```

4. Mr. Kim has to deliver refrigerators to N customers. From the office, he is going to visit all the customers and then return to his home. Each location of the office, his home, and the customers is given in the form of integer coordinates (x,y) (0≤x≤100, 0≤y≤100) . The distance between two arbitrary locations (x1, y1) and (x2, y2) is computed by |x1-x2| + |y1-y2|, where |x| denotes the absolute value of x; for instance, |3|=|-3|=3. The locations of the office, his home, and the customers are all distinct. You should plan an optimal way to visit all the N customers and return to his home among all the possibilities.

You are given the locations of the office, Mr. Kim’s home, and the customers; the number of the customers is in the range of 5 to 10. Write a program that, starting at the office, finds a (the) shortest path visiting all the customers and returning to his home. Your program only have to report the distance of a (the) shortest path.

Constraints

5≤N≤10. Each location (x,y) is in a bounded grid, 0≤x≤100, 0≤y≤100, and x, y are integers.

Input:

You are given 10 test cases. Each test case consists of two lines; the first line has N, the number of the customers, and the following line enumerates the locations of the office, Mr. Kim's home, and the customers in sequence. Each location consists of the coordinates (x,y), which is represented by 'x y'.

Output:

Output the 10 answers in 10 lines. Each line outputs the distance of a (the) shortest path. Each line looks like '#x answer' where x is the index of a test case. '#x' and 'answer' are separated by a space.

I/O Example ::::   Input (20 lines in total. In the first test case, the locations of the office and the home are (0, 0) and (100, 100) respectively, and the locations of the customers are (70, 40), (30, 10), (10, 5), (90, 70), (50, 20).)
5 Starting test case #1
0 0 100 100 70 40 30 10 10 5 90 70 50 20
6 Starting test case #2
88 81 85 80 19 22 31 15 27 29 30 10 20 26 5 14
10 Starting test case #3
39 9 97 61 35 93 62 64 96 39 36 36 9 59 59 96 61 7 64 43 43 58 1 36

Output (10 lines in total)
#1 200
#2 304
#3 366

There is dedicated Samsung software for coding test the question is given below:
There is one spaceship. X and Y co-odinate of source of spaceship and destination spaceship is given. There are N number of warmholes each warmhole has 5 values. First 2 values are starting co-ordinate of warmhole and after that value no. 3 and 4 represents ending co-ordinate of warmhole and last 5th value is represents cost to pass through this warmhole. Now these warmholes are bi-direction.
Now to go from (x1,y1) to (x2,y2) is abs(x1-x2)+abs(y1-y2).
The main problem here is to find minimum distance to reach spaceship from source to destination co-ordinate using any number of warm-hole. It is ok if you wont use any warmhole.

Arcesium

Tips :

- Try to attempt all the questions, if no negative marking
- Do all the coding problems. Sometimes, only those are called for next round who have done all the coding problems.

https://www.geeksforgeeks.org/arcesium-interview-experience-2018-intern-on-campus/
https://www.geeksforgeeks.org/arcesium-interview-experience-internship/
https://www.geeksforgeeks.org/arcesium-interview-experience-2018-intern/
https://www.geeksforgeeks.org/arcesium-interview-experience-set-7-campus-internship/
https://www.geeksforgeeks.org/arcesium-interview-experience-set-2/
https://www.geeksforgeeks.org/arcesium-interview-experience-set-4-internship/
https://www.geeksforgeeks.org/arcesium-interview-experience-set-6-campus-internship/
https://www.geeksforgeeks.org/arcesium-interview-experience-set-1/

ROUND 1 : Online Coding + Aptitude Test
Online coding and aptitude test on hacker rank.
The first round had 20 Aptitude MCQs (20 min) and 15 Technical MCQs (15 min) with +1 and -0.25 marking schemes. The MCQs covered topics the included – DSA, Operating Systems, C, C++, Java basics. After this, there were 2 coding questions (45 min). Minimum cut off was set for each section. One question was of the Game Theory and other was of Dynamic Programming.

Questions in The Coding Round: –

Q1) A and B play a game. They are given an array of positive numbers. Each player in his turn picks up 2 numbers from the array such that the difference of the numbers does not exist in the array. He then places the difference into the array too thus increase the array count by 1. Then the next player repeats the same process. The game continues till there are no 2 numbers such that difference does not exist in the array. The one who's not able to choose numbers loses. If A starts the game and the game proceeds optimally, find who'll win the game
Example : Input array : 2,5,3
A: 2,5,3,1 B: 2,5,3,1,4 A has no choice so B wins.

Q2) Given a string containing only lowercase alphabets, you have to convert it into a string such that it contains only vowels by doing minimum number of operations. In one operation, you could select a substring always starting from index 0, and move that substring forward or backward. Example of rolling forward or backward are given :
Rolling Forward
Input- axzf Let index chosen be 0 to 3 and moving it forward Output- byag Rolling Backward
Input – axze Let index chosen be 0 to 2 and moving it backwards Output- zwyd
124 people were eligible to give the test. Only 12 were shortlisted for round 2.

ROUND 1 :

Online test –

Had a total of 80 minutes and was conducted on hacker rank's code pair platform.

The section was divided into 3 parts,

1 Quantitative Aptitude 15 20

2 Technical Aptitude 15 15

3 Programming test 1 1 15

4 Programming test 2 1 30

Each question had equal weightage(they told in the interview)

There was no barrier of time in any section and you had the freedom to jump back and forth between the sections.

*Each wrong answer in Sections 1 and 2 carried a negative marking of 25% of the marks for that question and every question has four choices with only one correct answer.

The Quantitative had really good Questions and was tough for most for us, the technical was more about object-oriented

Questions in Java and C++ with few questions related to basic of C, stack, Binary Search Tree.The technical was decently easy if you are good at OOP(5-6 Code Snippets were given and we had to mark the appropriate answer).

1.There is a river of N unit and there are K stones across the river, each stone will be in one of the N units.Starting from 0th location your first jump will always be 1 unit (consider previous jump as L units), now your next subsequent jump can be either L + 1 or L – 1.Note few of the stones can be missing.

You have to output the sum of square of Jump which you'll take from each stone.

eg.let the river be of N = 5 units and there are 4 stones placed shown in array

[0, 1, 3, 4]

Solution –

1. The first jump from 0-1 is always fixed which will be 1 Unit.
2. Now for the next jump you have 2 options L+1 = 2 or L-1 = 0, we have to move forward from each location so it is not what we want, so we'll check if current location 1 + jump(L+1) i.e 1+2 = 3 is there in array or not, if yes we take a jump
3. Now from 3 we have 2 options (as last jump was 2 so we can jump 2+1 or 2-1 to reach next stone)

2-1 = 1 will make us reach to last stone 4 which we want

Now we have to output the sum of square of Jump's taken at each location

0^2 + 1^2 + 2^2 + 1^2 = 6

So if it is possible to reach the last stone output 6 else output -1

The actual solution is using DP but I tried Greedy way(couldn't pass all the testcases)

It was 90% similar to this problem – http://www.dsalgo.com/2016/01/jumping-frog-problem.html

2.You have been given a m*n chess board and there were black pawns on the board(represented by 1).You have to put your white pawn on all possible position's where it can be and the calculate how many kills are possible from each location and sum it up.You can only move straight (x+1, y) and for a kill diagonal move (x+1, y+1) & (x+1, y-1)

eg 3*3
0 0 0
0 1 0
1 1 1

-First row from 1st position (0, 0) => 2 kills, 2nd position (0, 1) => 0 kills, 3rd position (0, 2) => 2 kills

-Second row, from 1st position (1, 0) => 1 kill, 2nd position (1, 1) => Not possible, 3rd position (1, 2)=> 1 kill

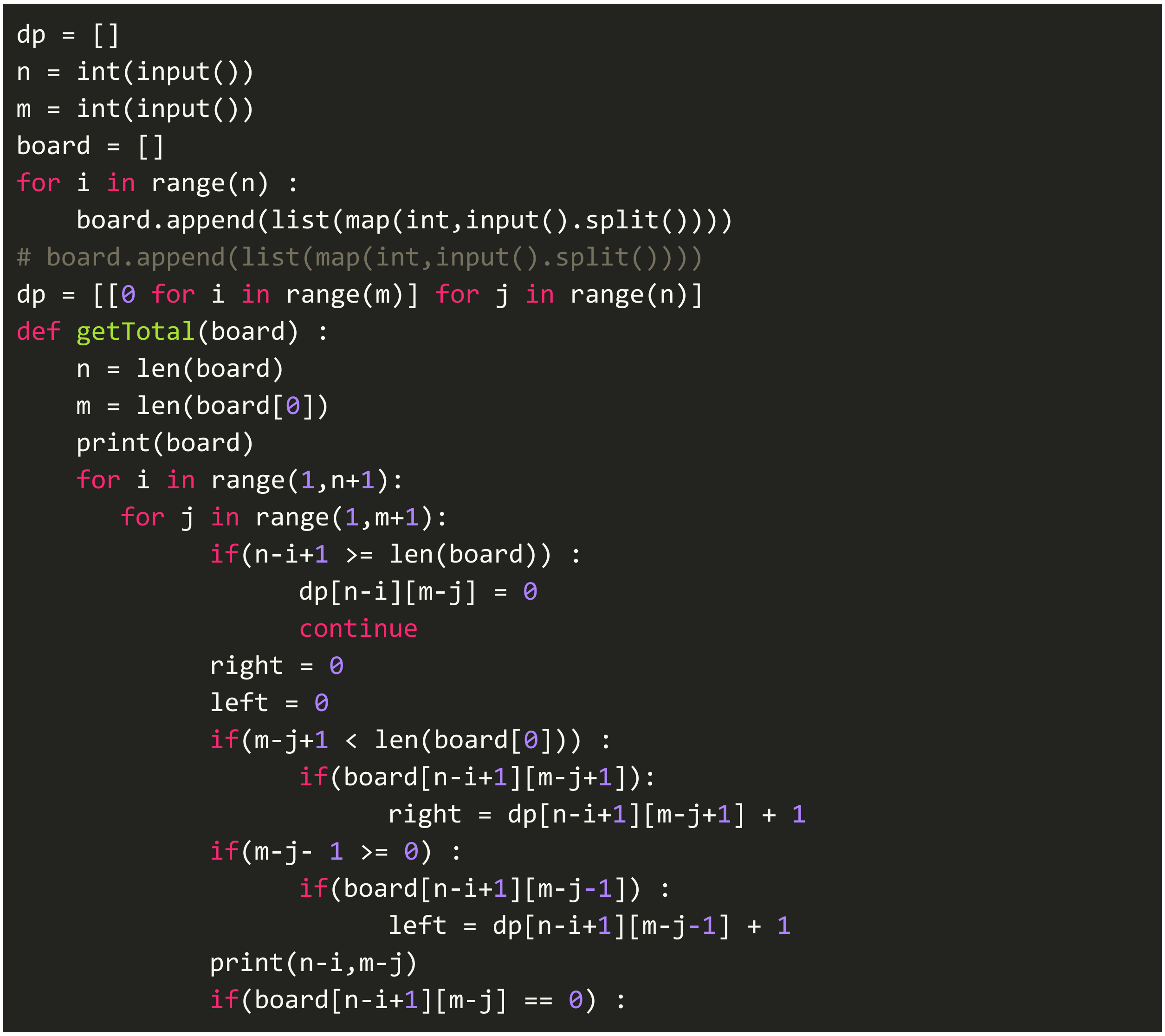
-Third row not possible to place as all are 1

At each step, you have to maximize the kills

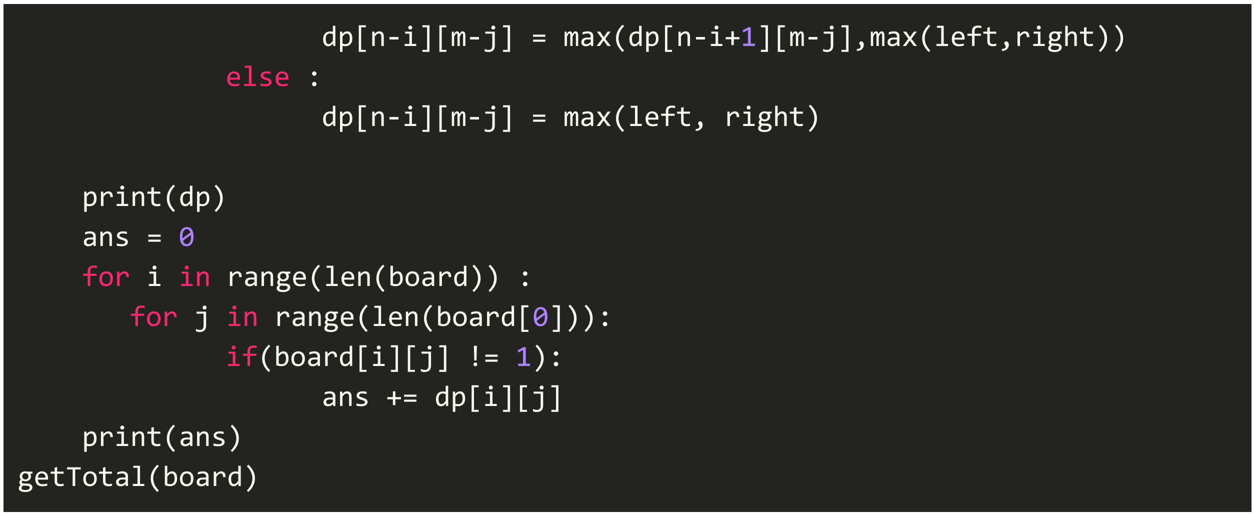
Output for the above question is – 6

-I had no time to solve this, even did not read the question properly

The link is to my friend's code for same problem using DP –
https://ide.geeksforgeeks.org/tf745kHTQI



```
dp = []
n = int(input())
m = int(input())
board = []
for i in range(n) :
    board.append(list(map(int,input().split())))
# board.append(list(map(int,input().split())))
dp = [[0 for i in range(m)] for j in range(n)]
def getTotal(board) :
    n = len(board)
    m = len(board[0])
    print(board)
    for i in range(1,n+1):
       for j in range(1,m+1):
             if(n-i+1 >= len(board)) :
                   dp[n-i][m-j] = 0
                   continue
             right = 0
             left = 0
             if(m-j+1 < len(board[0])) :
                   if(board[n-i+1][m-j+1]):
                         right = dp[n-i+1][m-j+1] + 1
             if(m-j- 1 >= 0) :
                   if(board[n-i+1][m-j-1]) :
                         left = dp[n-i+1][m-j-1] + 1
             print(n-i,m-j)
             if(board[n-i+1][m-j] == 0) :
```

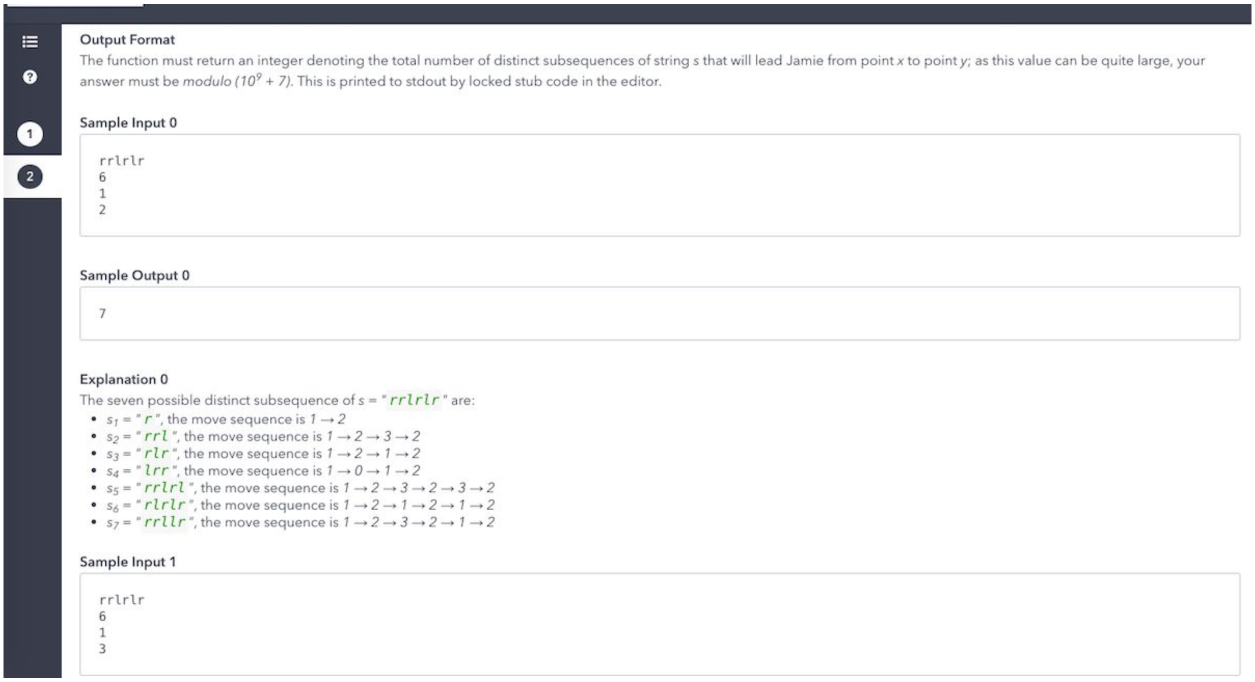


```
                   dp[n-i][m-j] = max(dp[n-i+1][m-j],max(left,right))
             else :
                   dp[n-i][m-j] = max(left, right)

    print(dp)
    ans = 0
    for i in range(len(board)) :
       for j in range(len(board[0])):
             if(board[i][j] != 1):
                   ans += dp[i][j]
    print(ans)
getTotal(board)
```

ROUND 1 : There were three sections.
Section1 : Quant (15 questions, 20 minutes) Questions were difficult to solve by most of the students. I solved just 1 question in this section.
Section2: Computer Science Aptitude (15 questions, 15 minutes) Consisted of many output questions in C, C++. Questions from OOP, OS and general CS were also there. I was able to attempt 8-9 questions.
Section3: Two programming questions (45 minutes).
Q1. A boat when has full fuel tank can cover *d* stops. There are n stops and it's given in an array that which of the stops have fuel station. Boat can be stopped at such stops to refill the tank. You have to minimize the number of stops to start from 1st station and reach the nth station. I was able to submit this question, passing all the test cases.

Q2. An inverted *apple tree* with two branches arising from each node is given. Basically a binary tree. An apple can be normal, nearly overhydrated or nearly underhydrated. Penalties for overhydrated and underhydrated apple are given. You have to water just one node in tree such that the sum of overhydrated and underhydrated penalties for the whole tree is minimized. Input for tree is given in the form of a parent array and another array denotes the status of each of the apple node as 0 (neutral), -1 (under hydrated) and 1 (overhydrated). My naive recursive code (without tree building) could pass only 5 out of 15 test cases.

Rivigo



Output Format

The function must return an integer denoting the total number of distinct subsequences of string *s* that will lead Jamie from point *x* to point *y*; as this value can be quite large, your answer must be *modulo* $(10^9 + 7)$. This is printed to stdout by locked stub code in the editor.

Sample Input 0

```
rrlrlr
6
1
2
```

Sample Output 0

```
7
```

Explanation 0

The seven possible distinct subsequence of s = "`rrlrlr`" are:

- s_1 = "`r`", the move sequence is $1 \rightarrow 2$
- s_2 = "`rrl`", the move sequence is $1 \rightarrow 2 \rightarrow 3 \rightarrow 2$
- s_3 = "`rlr`", the move sequence is $1 \rightarrow 2 \rightarrow 1 \rightarrow 2$
- s_4 = "`lrr`", the move sequence is $1 \rightarrow 0 \rightarrow 1 \rightarrow 2$
- s_5 = "`rrlrl`", the move sequence is $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 2$
- s_6 = "`rlrlr`", the move sequence is $1 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 2$
- s_7 = "`rrllr`", the move sequence is $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 2$

Sample Input 1

```
rrlrlr
6
1
3
```

Jamie is walking along a number line that starts at point 0 and ends at point n. She can move either one step to the left or one step to the right of her current location , with the exception that she cannot move left from point 0 or right from point n. In other words, if Jamie is standing at point i,she can move to either i-1 or i+1 as long as her destination exists in the inclusive range [0,n]. She has a string ,s , of movement instruction consisting of the letters 1 and r , where 1 is an instruction to move one step left and r is an instruction to move one step right.

Jamie followed the instructions in s one by one and in order .For Example if s='rrlr',she performs the following sequence of moves :one step right ->one step right ->one step left -> one step right .Jamie wants to move from point x to point y following some subsequence of string s instruction and wonders how many distinct possible subsequence of string s will get her from point x to point y. recall that a subsequence of a string is obtained by deleting zero or more characters from string .

it has four parameters

A String , s giving a sequence of eduction using the characters l(i.e. move left one unit) and r (i.e. move right one unit)

An integer n, denoting the length of the number line.

An integer x, denoting jamie's starting point on the number line

An integer y , denoting Jamie's ending point on the number line.

The function must return an integer denoting the total number of distinct subsequence of string s that will lead Jamie from point x to point y as this value can be quite large .

Sample Input
rrlrlr
6
1
2
output =7

```
#include<bits/stdc++.h>
using namespace std;
int dp[101][100][2];
int solve(string str, int idx, int flag, int s, int d, int n){
	if(s > n || s < 0)
		return 0;
	int len = str.length();
	if(idx >= len)
		return 0;
	int res = 0;

	if( (str[idx] == 'l'&&flag==0)||(str[idx]=='r'&&flag) )
	{
		int f = 1;
		if(str[idx] == 'l')
			f = -1;
res=solve(str,idx+1,flag,s+f,d,n)+solve(str,idx+1,1-flag,s+f,d,n);
		if(s + f == d)
			res++;
	}
	else{
		res = solve(str, idx + 1, flag, s, d, n);
	}
	dp[idx][s][flag] = res;
	return res;
}
int main(){
	string str;
	cin >> str;
	int len = str.length();
	int n;
	cin >> n;
```

```
      int s, d;
      cin >> s >> d;
      memset(dp, -1, sizeof(dp));
      int res = solve(str, 0 , 0, s, d, n) + solve(str, 0, 1, s, d, 
n);
      cout << res << endl;
}
```

Q) Cherry Pickup (Leetcode 741) :

In a N x N grid representing a field of cherries, each cell is one of three possible integers.

0 means the cell is empty, so you can pass through;
1 means the cell contains a cherry, that you can pick up and pass through;
-1 means the cell contains a thorn that blocks your way.

Your task is to collect maximum number of cherries possible by following the rules below:
Starting at the position (0, 0) and reaching (N-1, N-1) by moving right or down through valid path cells (cells with value 0 or 1);
After reaching (N-1, N-1), returning to (0, 0) by moving left or up through valid path cells;
When passing through a path cell containing a cherry, you pick it up and the cell becomes an empty cell (0);
If there is no valid path between (0, 0) and (N-1, N-1), then no cherries can be collected.

Example 1:
Input: grid = [[0, 1, -1],[1, 0, -1],[1, 1, 1]]
Output: 5
Explanation: The player started at (0, 0) and went down, down, right right to reach (2, 2).
4 cherries were picked up during this single trip, and the matrix becomes
[[0,1,-1],[0,0,-1],[0,0,0]].
Then, the player went left, up, up, left to return home, picking up one more cherry.
The total number of cherries picked up is 5, and this is the maximum possible.

```
class Solution {
      int[][][] memo;
      int[][] grid;
      int N;
      public int cherryPickup(int[][] grid) {
```

```
      this.grid = grid;
      N = grid.length;
      memo = new int[N][N][N];
      for (int[][] layer: memo)
            for (int[] row: layer)
                  Arrays.fill(row, Integer.MIN_VALUE);
      return Math.max(0, dp(0, 0, 0));
      }
      public int dp(int r1, int c1, int c2) {
      int r2 = r1 + c1 - c2;
      if (N == r1 || N == r2 || N == c1 || N == c2 ||
                  grid[r1][c1] == -1 || grid[r2][c2] == -1) {
            return -999999;
      } else if (r1 == N-1 && c1 == N-1) {
            return grid[r1][c1];
      } else if (memo[r1][c1][c2] != Integer.MIN_VALUE) {
            return memo[r1][c1][c2];
      } else {
            int ans = grid[r1][c1];
            if (c1 != c2) ans += grid[r2][c2];
            ans += Math.max(Math.max(dp(r1, c1+1, c2+1), dp(r1+1, c1,
c2+1)),
                              Math.max(dp(r1, c1+1, c2), dp(r1+1, c1,
c2)));
            memo[r1][c1][c2] = ans;
            return ans;
      }
      }
}
```

```
class Solution {
public:
    int cherryPickup(vector<vector<int>>& grid) {
        int n = grid.size();
        // dp holds maximum # of cherries two k-length paths can
pickup.
```

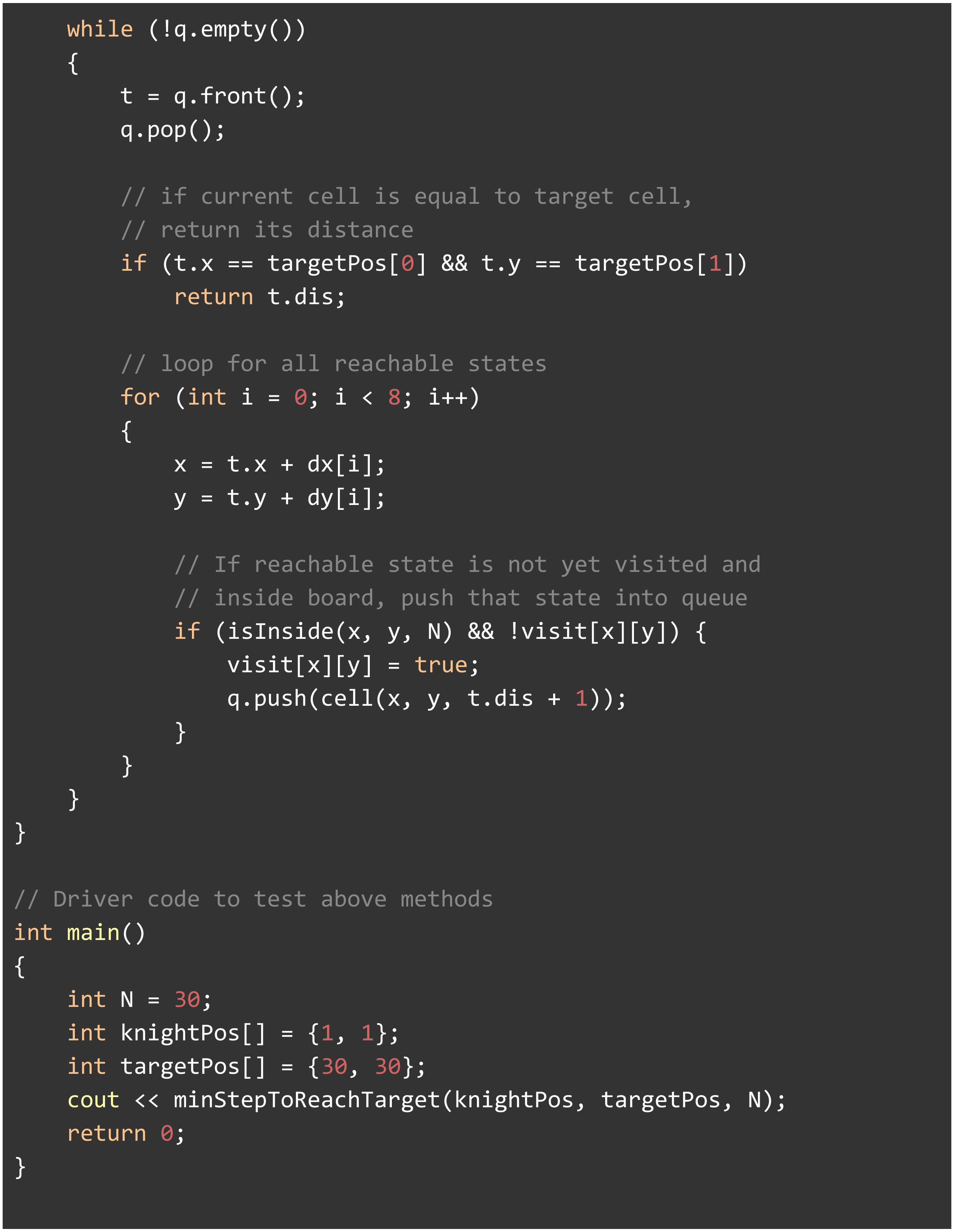
```
        // The two k-length paths arrive at (i, k - i) and (j, k -
j),
        // respectively.
        vector<vector<int>> dp(n, vector<int>(n, -1));

        dp[0][0] = grid[0][0]; // length k = 0

        // maxK: number of steps from (0, 0) to (n-1, n-1).
        const int maxK = 2 * (n - 1);

        for (int k = 1; k <= maxK; k++) { // for every length k
            vector<vector<int>> curr(n, vector<int>(n, -1));

            // one path of length k arrive at (i, k - i)
            for (int i = 0; i < n && i <= k; i++) {
                if ( k - i >= n) continue;
                // another path of length k arrive at (j, k - j)
                for (int j = 0; j < n && j <= k; j++) {
                    if (k - j >= n) continue;
                    if (grid[i][k - i] < 0 || grid[j][k - j] < 0) {
// keep away from thorns
                        continue;
                    }

                    int cherries = dp[i][j]; // # of cherries picked
up by the two (k-1)-length paths.

                    // See the figure below for an intuitive
understanding
                    if (i > 0) cherries = std::max(cherries, dp[i -
1][j]);

                    if (j > 0) cherries = std::max(cherries, dp[i][j
- 1]);

                    if (i > 0 && j > 0) cherries = std::max(cherries,
dp[i - 1][j - 1]);

                    // No viable way to arrive at (i, k - i)-(j,
k-j).
```

```
                    if (cherries < 0) continue;

                    // Pickup cherries at (i, k - i) and (j, k -j )
if i != j.
                    // Otherwise, pickup (i, k-i).
                    cherries += grid[i][k - i] + (i == j ? 0 :
grid[j][k - j]);

                    curr[i][j] = cherries;
                }
            }
            dp = std::move(curr);
        }

        return std::max(dp[n-1][n-1], 0);
    }
};
```

Q) Minimum steps to reach target by a Knight | Set 1
Given a square chessboard of N x N size, the position of Knight and position of a target is given. We need to find out minimum steps a Knight will take to reach the target position.

```
// C++ program to find minimum steps to reach to
// specific cell in minimum moves by Knight
#include <bits/stdc++.h>
using namespace std;

// structure for storing a cell's data
struct cell
{
    int x, y;
    int dis;
    cell() {}
    cell(int x, int y, int dis) : x(x), y(y), dis(dis) {}
```

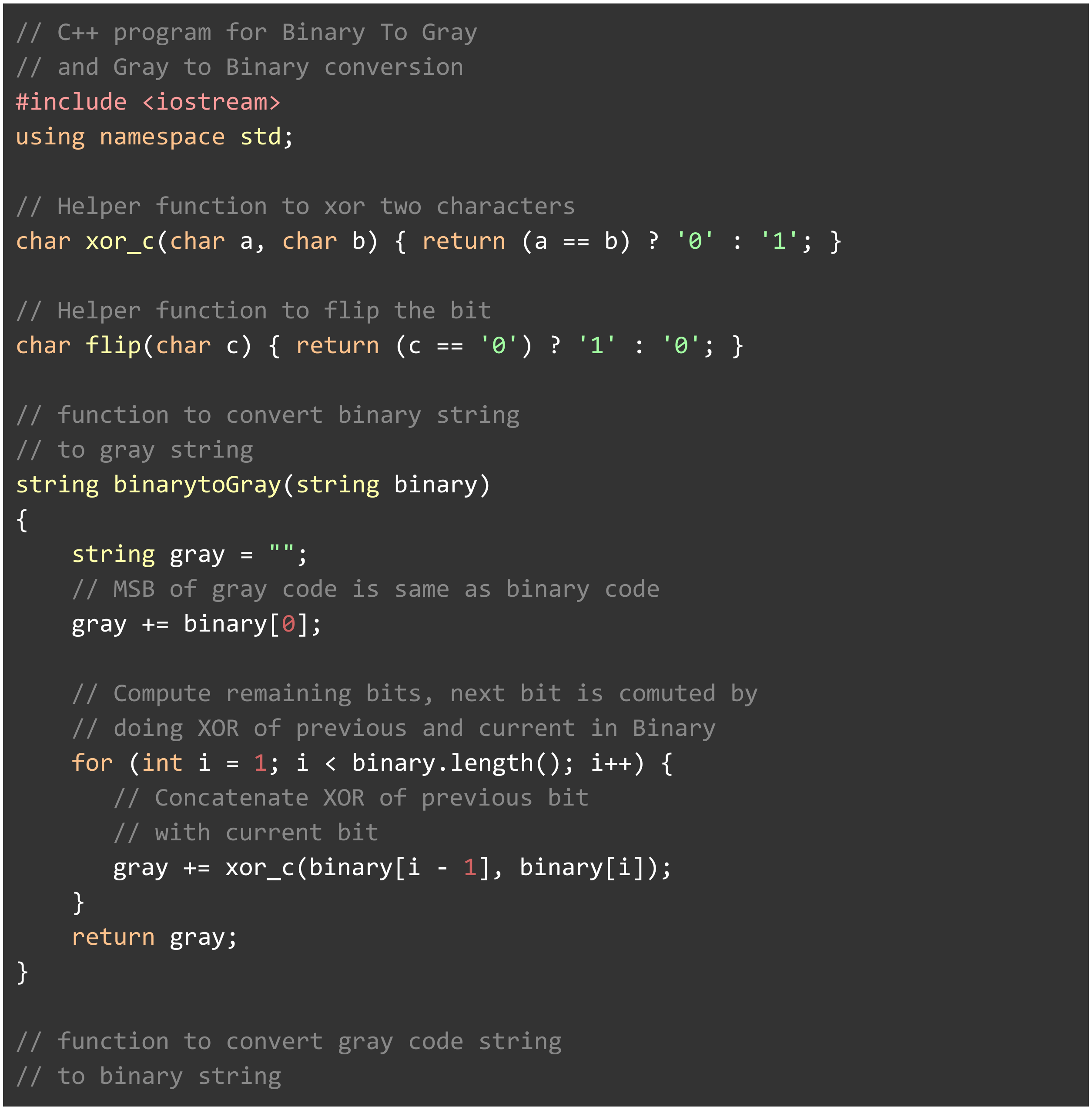
```
};

// Utility method returns true if (x, y) lies
// inside Board
bool isInside(int x, int y, int N)
{
    if (x >= 1 && x <= N && y >= 1 && y <= N)
        return true;
    return false;
}

// Method returns minimum step to reach target position
int minStepToReachTarget(int knightPos[], int targetPos[],
                                                int N)
{
    // x and y direction, where a knight can move
    int dx[] = {-2, -1, 1, 2, -2, -1, 1, 2};
    int dy[] = {-1, -2, -2, -1, 1, 2, 2, 1};

    // queue for storing states of knight in board
    queue<cell> q;

    // push starting position of knight with 0 distance
    q.push(cell(knightPos[0], knightPos[1], 0));

    cell t;
    int x, y;
    bool visit[N + 1][N + 1];

    // make all cell unvisited
    for (int i = 1; i <= N; i++)
        for (int j = 1; j <= N; j++)
            visit[i][j] = false;

    // visit starting state
    visit[knightPos[0]][knightPos[1]] = true;

    // loop until we have one element in queue
```



```
    while (!q.empty())
    {
        t = q.front();
        q.pop();

        // if current cell is equal to target cell,
        // return its distance
        if (t.x == targetPos[0] && t.y == targetPos[1])
            return t.dis;

        // loop for all reachable states
        for (int i = 0; i < 8; i++)
        {
            x = t.x + dx[i];
            y = t.y + dy[i];

            // If reachable state is not yet visited and
            // inside board, push that state into queue
            if (isInside(x, y, N) && !visit[x][y]) {
                visit[x][y] = true;
                q.push(cell(x, y, t.dis + 1));
            }
        }
    }
}

// Driver code to test above methods
int main()
{
    int N = 30;
    int knightPos[] = {1, 1};
    int targetPos[] = {30, 30};
    cout << minStepToReachTarget(knightPos, targetPos, N);
    return 0;
}
```

Minimum steps to reach target by a Knight | Set 2

Gray to Binary and Binary to Gray conversion : Binary Numbers is default way to store numbers, but in many applications binary numbers are difficult to use and a variation of binary numbers is needed. This is where Gray codes are very useful.

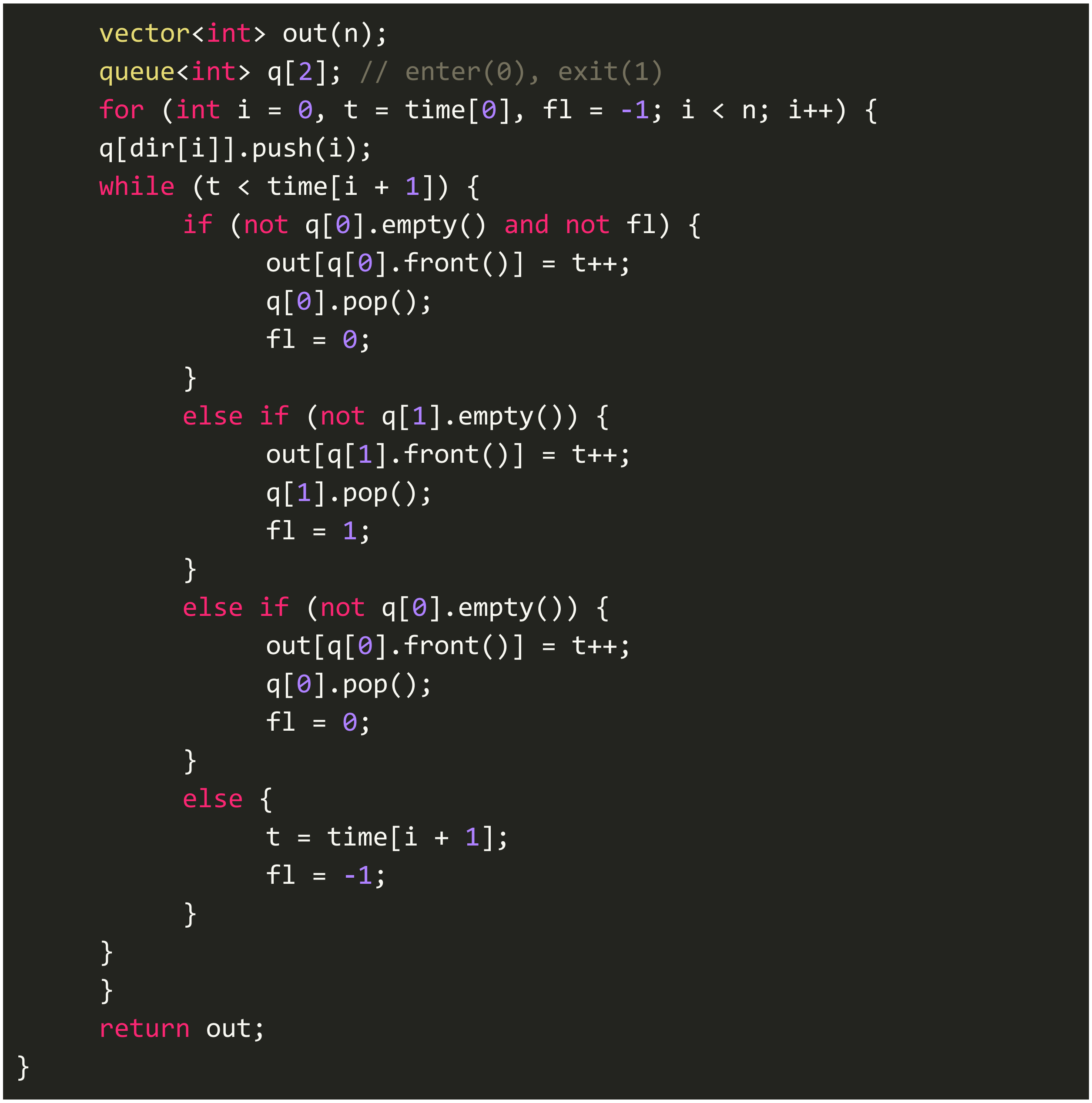


```
// C++ program for Binary To Gray
// and Gray to Binary conversion
#include <iostream>
using namespace std;

// Helper function to xor two characters
char xor_c(char a, char b) { return (a == b) ? '0' : '1'; }

// Helper function to flip the bit
char flip(char c) { return (c == '0') ? '1' : '0'; }

// function to convert binary string
// to gray string
string binarytoGray(string binary)
{
    string gray = "";
    // MSB of gray code is same as binary code
    gray += binary[0];

    // Compute remaining bits, next bit is comuted by
    // doing XOR of previous and current in Binary
    for (int i = 1; i < binary.length(); i++) {
       // Concatenate XOR of previous bit
       // with current bit
       gray += xor_c(binary[i - 1], binary[i]);
    }
    return gray;
}

// function to convert gray code string
// to binary string
```

```
string graytoBinary(string gray)
{
    string binary = "";

    // MSB of binary code is same as gray code
    binary += gray[0];

    // Compute remaining bits
    for (int i = 1; i < gray.length(); i++) {
       // If current bit is 0, concatenate
       // previous bit
       if (gray[i] == '0')
             binary += binary[i - 1];

       // Else, concatenate invert of
       // previous bit
       else
             binary += flip(binary[i - 1]);
    }
    return binary;
}

// Driver program to test above functions
int main()
{
    string binary = "01001";
    cout << "Gray code of " << binary << " is "
       << binarytoGray(binary) << endl;

    string gray = "01101";
    cout << "Binary code of " << gray << " is "
       << graytoBinary(gray) << endl;
    return 0;
}
```

Turnstile

```
vector<int> getTimeStamps(vector<int> time, vector<int> dir) {
      int n = time.size();
      time.push_back(1E9 + 1E6);
```



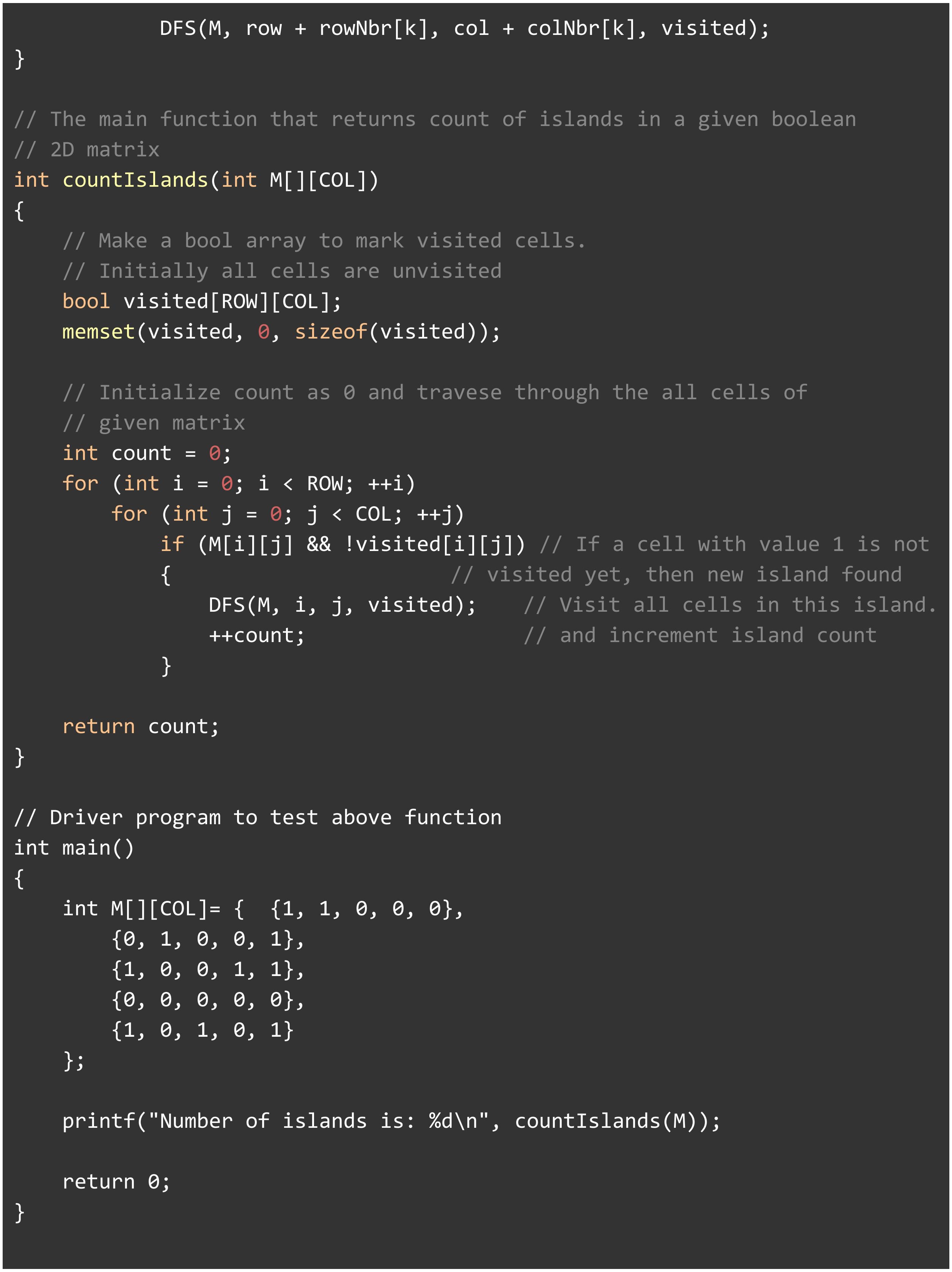
```
    vector<int> out(n);
    queue<int> q[2]; // enter(0), exit(1)
    for (int i = 0, t = time[0], fl = -1; i < n; i++) {
    q[dir[i]].push(i);
    while (t < time[i + 1]) {
        if (not q[0].empty() and not fl) {
            out[q[0].front()] = t++;
            q[0].pop();
            fl = 0;
        }
        else if (not q[1].empty()) {
            out[q[1].front()] = t++;
            q[1].pop();
            fl = 1;
        }
        else if (not q[0].empty()) {
            out[q[0].front()] = t++;
            q[0].pop();
            fl = 0;
        }
        else {
            t = time[i + 1];
            fl = -1;
        }
    }
    }
    return out;
}
```

Cohesity 2018

1. https://imgur.com/gallery/M0RKXgJ

2. https://www.geeksforgeeks.org/find-number-of-islands/
   Solution :
   // Program to count islands in boolean 2D matrix

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

#define ROW 5
#define COL 5

// A function to check if a given cell (row, col) can be included in DFS
int isSafe(int M[][COL], int row, int col, bool visited[][COL])
{
    // row number is in range, column number is in range and value is 1
    // and not yet visited
    return (row >= 0) && (row < ROW) &&
           (col >= 0) && (col < COL) &&
           (M[row][col] && !visited[row][col]);
}

// A utility function to do DFS for a 2D boolean matrix. It only considers
// the 8 neighbours as adjacent vertices
void DFS(int M[][COL], int row, int col, bool visited[][COL])
{
    // These arrays are used to get row and column numbers of 8 neighbours
    // of a given cell
    static int rowNbr[] = {-1, -1, -1,  0, 0,  1, 1, 1};
    static int colNbr[] = {-1,  0,  1, -1, 1, -1, 0, 1};

    // Mark this cell as visited
    visited[row][col] = true;

    // Recur for all connected neighbours
    for (int k = 0; k < 8; ++k)
        if (isSafe(M, row + rowNbr[k], col + colNbr[k], visited) )
```



```
            DFS(M, row + rowNbr[k], col + colNbr[k], visited);
}

// The main function that returns count of islands in a given boolean
// 2D matrix
int countIslands(int M[][COL])
{
    // Make a bool array to mark visited cells.
    // Initially all cells are unvisited
    bool visited[ROW][COL];
    memset(visited, 0, sizeof(visited));

    // Initialize count as 0 and travese through the all cells of
    // given matrix
    int count = 0;
    for (int i = 0; i < ROW; ++i)
        for (int j = 0; j < COL; ++j)
            if (M[i][j] && !visited[i][j]) // If a cell with value 1 is not
            {                       // visited yet, then new island found
                DFS(M, i, j, visited);    // Visit all cells in this island.
                ++count;                  // and increment island count
            }

    return count;
}

// Driver program to test above function
int main()
{
    int M[][COL]= {  {1, 1, 0, 0, 0},
        {0, 1, 0, 0, 1},
        {1, 0, 0, 1, 1},
        {0, 0, 0, 0, 0},
        {1, 0, 1, 0, 1}
    };

    printf("Number of islands is: %d\n", countIslands(M));

    return 0;
}
```

3. https://www.hackerrank.com/contests/codegamblefighton/challenges/k-most-frequent/problem
   Solution :

```
#include<bits/stdc++.h>
using namespace std;


bool cmp(const pair<int,int> &a, const pair<int,int> &b){
      if(a.first==b.first) return a.second<b.second;
      return (a.first>b.first);
}

bool comp(pair<int, int>a, pair<int, int>b){
      if(a.first==0) return 1;
      if(b.first==0) return 1;
      return a.first==b.first;
}

int main(){
      int T;
      cin>>T;


      while(T--){
      set<int> setmera;
      pair<int, int> temp[26];
      char str[100000];
      char ch;
      int n,k,x;

      cin>>n;
      cin>>k;
      for(int i=0; i<n; i++)
            cin>>str[i];

      if(k>26)cout<<"NONE"<<endl;
      else{
            for(int i=0; i<26; i++){
                  temp[i].first=0;
                  temp[i].second=i;
            }


            for(int i=0; i<n; i++) temp[str[i]-'a'].first++;

            sort(temp, temp+26, cmp);

            auto ptr = unique(temp, temp+26, comp);

            if(k<=ptr-temp){
                  char ch='a';
                  ch+=temp[k-1].second;
                  cout<<ch<<endl;
            }
            else cout<<"NONE"<<endl;

      }
      }
      return 0;
}
```

4.https://www.geeksforgeeks.org/maximum-sum-in-circular-array-such-that-no-two-elements-are-adjacent/ (solution mujhe clear nhi hua iska)
// CPP program to find maximum sum in a circular array such that no elements are adjacent in the sum.

```
#include <bits/stdc++.h>
using namespace std;

// Function to calculate the sum
// from 0th position to(n-2)th position
int maxSum1(int arr[], int n)
{
    int dp[n];
    int maxi = 0;

    for (int i = 0; i < n - 1; i++) {
        // copy the element of original array to dp[]
        dp[i] = arr[i];

        // find the maximum element in the array
        if (maxi < arr[i])
            maxi = arr[i];
    }

    // start from 2nd to n-1th pos
    for (int i = 2; i < n - 1; i++) {

        // traverse for all pairs
        // bottom-up approach
        for (int j = 0; j < i - 1; j++) {

            // dp-condition
            if (dp[i] < dp[j] + arr[i]) {
                dp[i] = dp[j] + arr[i];

                // find maximum sum
                if (maxi < dp[i])
                    maxi = dp[i];
            }
        }
    }

    // return the maximum
    return maxi;
}

// Function to find the maximum sum
// from 1st position to n-1-th position
int maxSum2(int arr[], int n)
{
    int dp[n];
    int maxi = 0;

    for (int i = 1; i < n; i++) {
        dp[i] = arr[i];

        if (maxi < arr[i])
```

```
            maxi = arr[i];
    }

    // Traverse from third to n-th pos
    for (int i = 3; i < n; i++) {

        // bootom-up approach
        for (int j = 1; j < i - 1; j++) {

            // dp condition
            if (dp[i] < arr[i] + dp[j]) {
                dp[i] = arr[i] + dp[j];

                // find max sum
                if (maxi < dp[i])
                    maxi = dp[i];
            }
        }
    }

    // return max
    return maxi;
}

int findMaxSum(int arr[], int n)
{
   return max(maxSum1(arr, n), maxSum2(arr, n));
}

// Driver Code
int main()
{
    int arr[] = { 1, 2, 3, 1 };
    int n = sizeof(arr)/sizeof(arr[0]);
    cout << findMaxSum(arr, n);
    return 0;
}
```

5. From 0 make X such that you can add, subtract and double. Cost of adding and subtracting is A whereas cost of doubling is B. Find minimum cost. (35 M)

I/P - X=4, A=1, B=1 O/P- 3 Explanation 0->1->2->4 #each link is of cost 1.

Solution : (Similar)
https://www.geeksforgeeks.org/minimum-number-operation-required-convert-number-x-y/

Two questions. 1 question for 50 marks and other for 20 marks.
6. Given a 2D grid, starting point, return minimum distance to reach the boundary of the grid. Some of the cells in the grid have obstacles. (Sol. - DFS/BFS) - 50 marks (Sre typ ka h)

MICROSOFT (https://www.geeksforgeeks.org/microsofts-asked-interview-questions/)

1. Group Fly Round: Students were divided into groups of 3 or 4 and were assigned a mentor. Everyone was asked the same question and was expected to write a clear and neat C/C++ code with comments wherever possible.
The question was that an 1-D array contained N*N elements. Assuming that the N*N elements form a matrix, you have to rotate the matrix in-place.

2. Group fly round:
Remove and replace the character ‘c’ from a given input string by double characters “**”
Input: “calcic”   Output: **al**i**

Given a binary tree whose structure is as below

```
Class BSTspecial{
public:
  BSTspecial* parent = NULL;
  BSTspecial* left = NULL;
  BSTspecial* right = NULL;
  int data;
  BSTspecial(int data){
    this->data = data;
} };
```

Given a node (note it can be either of nodes of the tree whether it is root or not) you need to find its immediate right sibling/cousin if any or return NULL if not present.
Input: 1 2 3 4 5 -1 6 -1 -1 -1 -1 -1 -1(level order input)
For node ‘5’ answer is ‘6’
For node ‘4’ answer is ‘5’
For node ‘6’ answer is -1
For node ‘1’ answer is -1

3. **Group fly round:** There were two questions given in this round. The code was to be written on paper in 45 mins.

1. Check if string 1 is a rotated version of string 2,
2. A variation of the word break problem. Given a dictionary of words we had to find out the longest word in the dictionary that could be formed from the rest of the words.

Note that you have only 45 mins. They expect you to write the code for both the problems. I had written a dynamic programming solution for the second question but wrote a brute force for the first (O(n^2)). Note that the first question could also be solved in O(n) using kmp but I would not have had the time to complete both questions if I tried the kmp approach. A fellow student who tried the kmp could not write complete code and was not shortlisted for further rounds.
A very important point in the group fly is to interact with the interviewers. They are very nice and helpful. Explaining your solution makes it easier for them to evaluate your solution later.

4. **Group Test :** In contrast to their traditional second round i.e. group fly. Second round was a group test. All students were asked to write the code for two question on paper in one hour. The questions were given in 30 min gap. Unlike group fly round you just have to write the code.
Question 1: https://www.geeksforgeeks.org/connect-nodes-at-same-level/
(instead of nextRight we have to link nextLeft)
Question 2: https://www.geeksforgeeks.org/cutting-a-rod-dp-13/

Tips: Your code must be neat and clean.

1. Use proper indentation.
2. Use as many comments as possible.
3. Dry run your code
4. Write time and space complexity of your approach.
5. your approach must be easily understood.
6.

Trust me, they evaluate the candidates on above points only. 16 Candidates were selected in this round.

5. **Group fly :** there were 2 questions :

1. Infix to postfix
2. connect nodes at same level in a binary tree

6. **(Group Fly) 30 mins**

Question – Given a BST print out all root to leaf paths whose sum equal to a given number. Also, write a full program which includes the construction of the tree and then find the paths.

Answer -This was a written round and there was one question for which we had to write the full program. It is very important that you ask questions from the instructor about the problem, to have the full understanding of the problem and what they actually need. Don’t be in a hurry as mere solving is not important, your solution must be correct and optimized so it is highly advisable to tell out your approach to your mentor. I would say the deciding factors in this round generally boils down to the clarity you have about the problem and the things demanded in the solution, sometimes they even ask to write down the algorithm, measure the time complexity, give out proper test cases and then write a !! CLEAN CODE!!!. (Yes handwriting and clarity matters or else they might even not read your paper, they are humans too)

7. **group fly round.**
There were 2 questions:
1.) Resolve the unix path. (I did this using a stack)
2.) No of encodings possible(based on DP)

8. **Group Flyer) :** 2 questions are given and you have to discuss the approach with the mentor and write the optimised approach on a paper.
Question 1: https://www.geeksforgeeks.org/word-break-problem-using-backtracking/
Variation of the above problem was given, one has to find the minimum number of words which one can use for making a given sentence.
Question 2: Students of different heights are attending an assembly. The problem is that if a student has less or equal height than the student standing above him then he/she cannot see the assembly. Task was to find the minimum number of students randomly such that maximum number of students can see the assembly.
Ex: 9 1 2 3 1 5
output : 2 students i.e. 9 and 1 has to be removed so that 4 students can see the assembly.
I saw a pattern that its an implementation of LIS.
Advise: Always discuss your approach with the mentor as much as possible and write a very neat and clean code on the paper. Always try to give the time and space complexity of the solution you are providing.This round is a group flyer but you have to code individually.

9. **Group fly** (30 MINUTES) This round was pen and paper round.All students were given a question and we need to write algorithm as well as code on paper. write neat code and properly explain your algorithm. question is given below:
https://www.geeksforgeeks.org/perfect-sum-problem-print-subsets-given-sum/

10. **Group Fly Round :** 2 questions were given to be solved in 2 hours. Both were pretty standard dynamic programming questions.
1. Remove min no of elements such that the resulting sequence is in strictly increasing order.
2. Word Break problem. Our problem was a slight variation of the problem in the link. We had to find if the string forms a meaningful sentence with **fewest** words possible.

11. **Group Fly Round, 45 Minutes, 2 coding questions, Written Round (~20/65 shortlisted)**

This was a written round and we were expected to write fully functional code, without any bugs and errors. Library functions used, if any, had to be explained and if possible, code for that too (not that rigorous). Pseudo code and algorithms were also allowed. All the assumptions made had to be explained as well.

1. Print the nth line of the pattern given
   - 1
   - 11
   - 21
   - 1211
   - 111221
   - 312211
   - 13112221
   - …

https://www.geeksforgeeks.org/look-and-say-sequence/
1. 2Given an array of numbers, arrange them in a way that yields the largest value.
https://www.geeksforgeeks.org/given-an-array-of-numbers-arrange-the-numbers-to-form-the-biggest-number/