Name :- Ayush Rath
University Roll no - 2023042

1)
```
#include <assert.h>
#include <ctype.h>
#include <limits.h>
#include <math.h>
#include <stdbool.h>
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char* readline();
char* ltrim(char*);
char* rtrim(char*);
char** split_string(char*);

int parse_int(char*);

/*
 * Complete the 'minimumAverage' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts 2D_INTEGER_ARRAY customers
 as parameter.
 */
int minimumAverage(int customers_rows, int customers
_columns, int** customers) {
```

Ayush
22/06/21

```
}
int main()
{
  FILE* fptr = fopen(getenv("OUTPUT_PATH"),"w");
  int n = parse_int(ltrim(rtrim(readline())));
  int** customers = malloc(n* sizeof(int*));
  for(int i=0; i<n; i++) {
    *(customers+i) = malloc(2*(sizeof(int)));
    char** customer_item_temp = split_string(rtrim
    (readline()));
  for(int j=0; j<2; j++){
  int customer_item = parse_int(*(customer_item
  _temp+j));
    ((customers+i)+j) = customer_item;
    }
  }
  int result = minimumAverage(n, 2, customers);
  fprintf(fptr, "%d\n", result);
  fclose(fptr);
  return 0;
}
char* readline(){
```

22/06/21

```
size_t alloc_length = 1024;
size_t data_length = 0;

char* data = malloc(alloc_length);
while (true) {
    char* cursor = data + data_length;
    char* line = fgets(cursor, alloc_length - data_length, stdin);
    if (!line) {
        break;
    }
    data_length += strlen(cursor);
    if (data_length < alloc_length - 1 || data[data_length - 1] == '\n') {
        break;
    }
    alloc_length <<= 1;
    data = realloc(data, alloc_length);
    if (!data) {
        data = "10";
        break;
    }
}
```

Dyuthi
22/06/21

```
if (data[data_length - 1] == '\n') {
  data[data_length-1] = '\0';

  data = realloc(data, data_length);
  if (!data) {
    data = '\0';
  }
 } else {
   data[data_length] = '\0';
  }
}
 return data;

}
char* ltrim(char* str) {
 if (!str) {
  return '\0';
}
if (!*str) {
  return str;
}
while (*str != '\0' && isspace(*str)) {
  str++;
}
```

Dyush
22/06/21

```
    return str;
}
char* trim(char* str) {
    if (!str) {
        return "";
    }
    if (!*str) {
        return str;
    }
    char* end = str + strlen(str) - 1;
    while (end >= str && isspace(*end))
    {
        end--;
    }
    *(end + 1) = '\0';
    return str;
}
char** split_string(char* str) {
    char** splits = NULL;
    char* token = strtok(str, " ");
    int spaces = 0;
```

Piyush
22/06/21

```
while (token){
  splits = realloc (splits, sizeof (char*)* ++spaces);
  if (!splits) {
    return splits;
  }
  splits [spaces - 1] = token;
  token = strtok (NULL, " ");
}
  return splits;
}
int parse_int (char* str){
  char* endptr;
int value = strtol (str, &endptr, 10);
if (endptr == str || *endptr != '\0') {
  exit (EXIT_FAILURE);
}
  return value;
}
```