Khushboo Upadhyay
20052088.
BSC IT.

Ques 1.

Ans.

```c
#include <stdio.h>
#include <ctype.h>
#include <limits.h>
#include <math.h>
#include <stdbool.h>
#include <stddef.h>
#include <stdin.h>
#include <stdlib.h>
#include <string.h>
char* readline();
char* ltrim(char*);
char* rtrim(char*);
int parse_int(char*);
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*(size of (int)));
    char** customers_item_temp = split_string(rtrim(readline()))
    for (int j=0; j<2; j++)
    { int customers_item = parse_int(*(customers_item_
    temp + j));
```

```c
((customers + i) + j) = customers - item; }}
int result = minimumAverage (n, 2, customers);
fprintf (fptr, "%d\n", result);
fclose (fptr);
return 0;
}
char * readline () {
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; }
if(data_length < alloc_length -1 || data[data_length -1] == '\n')
    { break; }
alloc_length <<= 1;
data = realloc (data, alloc_length);

if (!data) {
data = '\0';

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

```c
else
{ data = realloc (data, data - length + 1);
 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++;
 }
return str;
}
char * rtrim (char * str) {
if (! str) {
 return '\0';
 }
if (!* str) {
 return str;
 }
char * end = str + strlen (str) - 1;
while (end >= str && isspace (* end)) {
 end -- ; }
```

```c
    *(end+1) = '\0';
    return str;
}
char ** split-string ( char * str)
{   char ** splits = NULL;
    char * token = strtok (str, " ");
    int spaces = 0;
    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;
}
```