classmate
Date
Page

```
#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*);

int minimumAverage (int customers_rows, int
customers_colums, int ** customers) {

}.

int main()

{
    FILE * fptr = fopen (getenv ("OUTPUT_PATH").
    , "w");

int n= parse_int (ltrim(rtrim(readline())));
```

[illegible]

```
int ** customers = malloc (n * size of (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));

       * (* (customers + i) + j) = customers_item;

  }.

}.

  int result = minimum Average (n, 2, customers);

   fprintf (fptr, "%d\n", result);

   f close (fptr);

    return 0;

}.

char * readline () {
```

[illegible]

Date
Page

```
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 (datalength < alloc_length - 1 || data [data_length
                  - 1] == '\n') {
        break;

  }

    alloc_length <<= 1;

  data = realloc (data, alloc_length);

  if (!data) {

      data = '\0';
```

[illegible]

```
            break;
        }.
    }.
    if (data [data_length - 1] == '\n') {.
        data [data_length - 1] = '\0';
      data =  realloc (data, data_length);
      if (!data) {
          data = '\0';
      }.
    } else {.
        data = realloc (data, data length + 1);
        if (!data) {
            data = '\0';
        } else {.
            data [data_length] = '\0';
        }.
    }.
```

[illegible]

```
    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) {
```

[illegible]

classmate
Date
Page

```
    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;

  while (token) {

    splits = realloc (splits, sizeof ( Char *) * ++spa
-ces.);

   if (!splits) {
```

[illegible]

```
    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 ;

   }.
```

[illegible]

Date ______
Page ______

Sample input # 00

3
0 3
1 9
2 6

Sample output # 00

9

Sample input # 01

3
0 3
1 9
2 5

Sample output # 01

8.

Explanation # 01.

Let's call the person ordering at time = 0 as A, time = 1 as B & time = 2 as C. By delivering pizza for A, C & B we get the minimum average wait time to be.
(3 + 6 + 16)/3 = 25/3 = 8.33