Gebze Technical University
Department of Computer Engineering

CSE 344 - HOMEWORK 4 REPORT

THREADS

Student:
Ahmet Melih YANALAK
Advisor:
Prof. Dr. Erchan APTOULA

20-05-2021

Contents

1 Definition Of The Project

The aim of this project is simulating a workflow of a student who hires other students from different universities.Producer will get the homeworks and put them in a queue.Main Thread will get the homeworks from queue and assign them to one of he available students which they will do for money that specified before the flow starts.

2 Visualizing the Structure

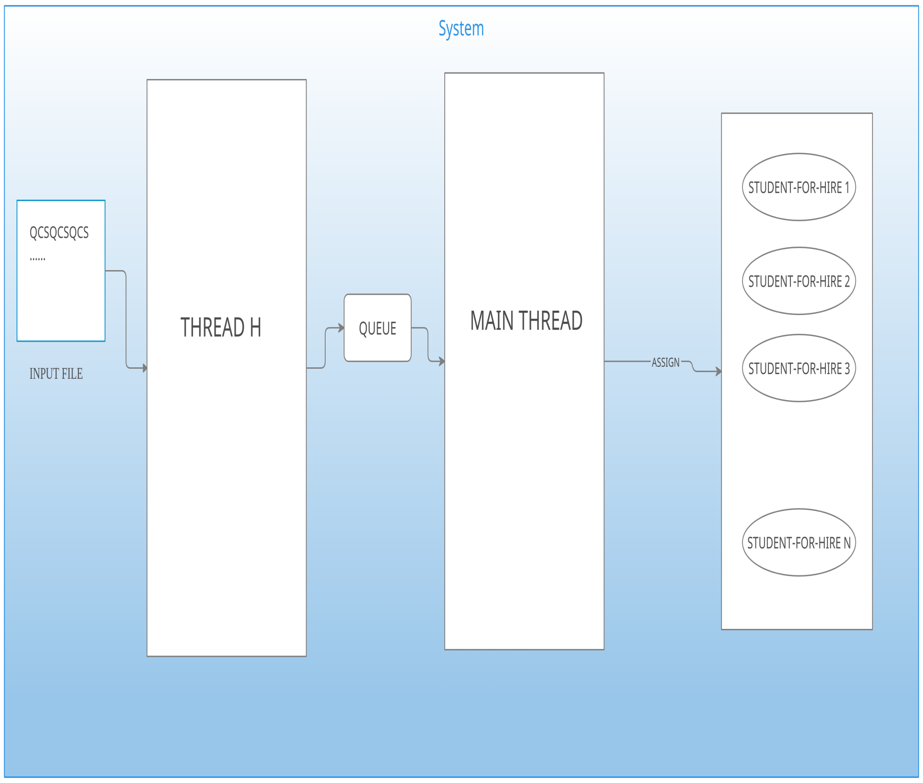


Figure 1.1: Structure of the program

In order to start the flow, Producer Thread will produce homeworks to queue,meanwhile main thread taking the homeworks from the queue and according to homework type it selects the most suitable student available.Main threads assign the homework to that student and decrease the amount of money that producer has.If there are not room in the queue then producer waits until main thread consuming the homeworks from the queue.

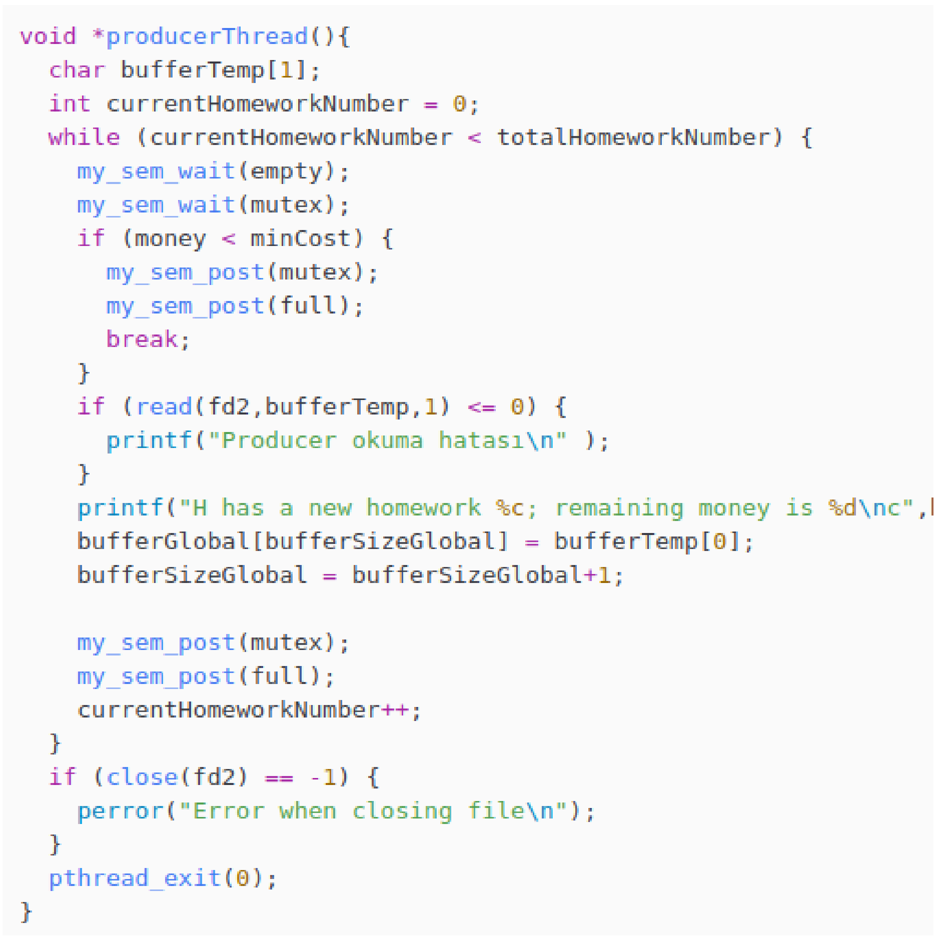
3 Threads

3.1 Producer Thread Design and Implementation

Producer thread should read the ASCII Input file one byte at a time and add those values to queue.After reading one character which can be 'Q', 'C' or 'S', the character will be putted to the queue and semaphore named "full" will be increased.This operations happens in critical region,thus a semaphore called "mutex" was needed for achieving this issue.

The termination condition for producer thread is reading the all of the homeworks or running out of money.

NOTE: This figure is not the latest version.It only used to explain the concept(Signal handling part is not shown yet for the sake of simplicity).



```
void *producerThread(){
  char bufferTemp[1];
  int currentHomeworkNumber = 0;
  while (currentHomeworkNumber < totalHomeworkNumber) {
    my_sem_wait(empty);
    my_sem_wait(mutex);
    if (money < minCost) {
      my_sem_post(mutex);
      my_sem_post(full);
      break;
    }
    if (read(fd2,bufferTemp,1) <= 0) {
      printf("Producer okuma hatası\n" );
    }
    printf("H has a new homework %c; remaining money is %d\nc",
    bufferGlobal[bufferSizeGlobal] = bufferTemp[0];
    bufferSizeGlobal = bufferSizeGlobal+1;

    my_sem_post(mutex);
    my_sem_post(full);
    currentHomeworkNumber++;
  }
  if (close(fd2) == -1) {
    perror("Error when closing file\n");
  }
  pthread_exit(0);
}
```

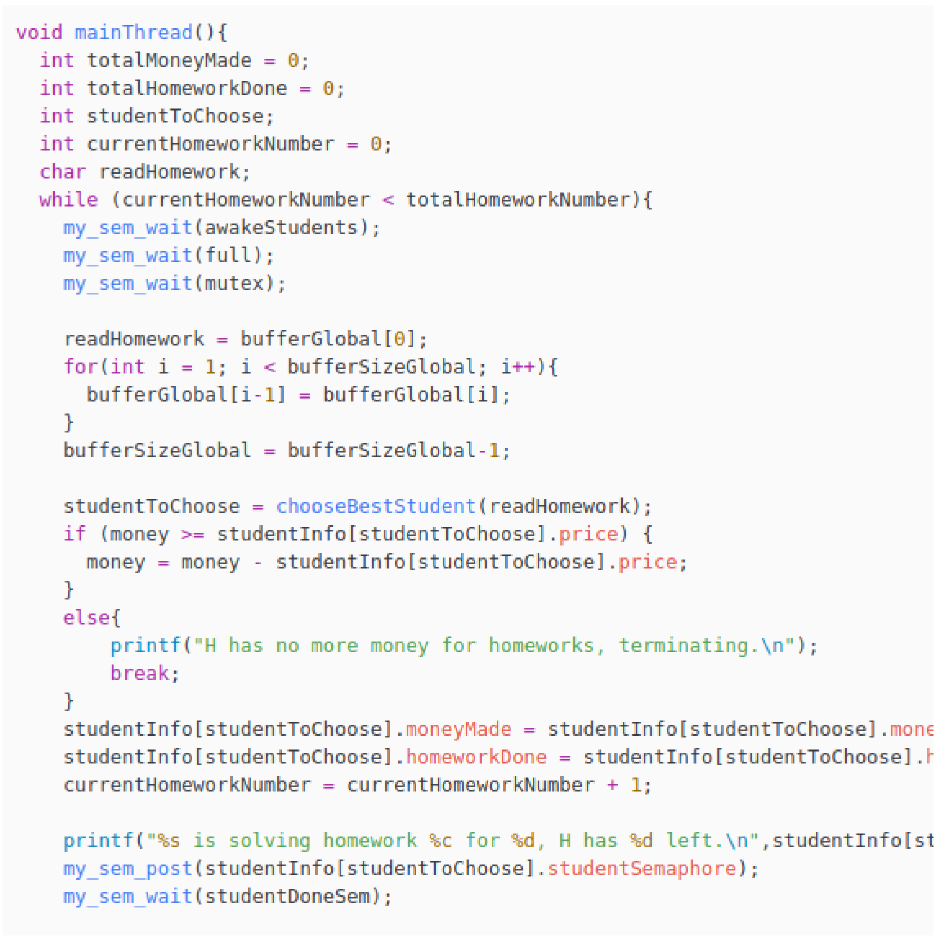
Figure 1.2: Implementation of the Producer Thread

3.2 Main Thread Design and Implementation

Main Thread should wait until there isat least one homework in the buffer. The "full" semaphore is used in order to provide the synchronization.If there are not sufficient number of homeworks then main thread is sleeping.After producer providing one of the homework types and posting full semaphore, main thread will wait for mutex to be unlocked and enter the critical region.

In the critical region main thread knows that it has at least one homework that student-for-hires need,main thread invites the student-for-hires which is the most suitable by communicating via semaphores for each student thread. Each citizen has their own semaphore in order to figure out if that process itself is chosen.

NOTE: This figure is not the latest version.It only used to explain the concept(Signal handling part is not shown yet for the sake of simplicity).



```
void mainThread(){
  int totalMoneyMade = 0;
  int totalHomeworkDone = 0;
  int studentToChoose;
  int currentHomeworkNumber = 0;
  char readHomework;
  while (currentHomeworkNumber < totalHomeworkNumber){
    my_sem_wait(awakeStudents);
    my_sem_wait(full);
    my_sem_wait(mutex);

    readHomework = bufferGlobal[0];
    for(int i = 1; i < bufferSizeGlobal; i++){
      bufferGlobal[i-1] = bufferGlobal[i];
    }
    bufferSizeGlobal = bufferSizeGlobal-1;

    studentToChoose = chooseBestStudent(readHomework);
    if (money >= studentInfo[studentToChoose].price) {
      money = money - studentInfo[studentToChoose].price;
    }
    else{
        printf("H has no more money for homeworks, terminating.\n");
        break;
    }
    studentInfo[studentToChoose].moneyMade = studentInfo[studentToChoose].mone
    studentInfo[studentToChoose].homeworkDone = studentInfo[studentToChoose].h
    currentHomeworkNumber = currentHomeworkNumber + 1;

    printf("%s is solving homework %c for %d, H has %d left.\n",studentInfo[st
    my_sem_post(studentInfo[studentToChoose].studentSemaphore);
    my_sem_wait(studentDoneSem);
```

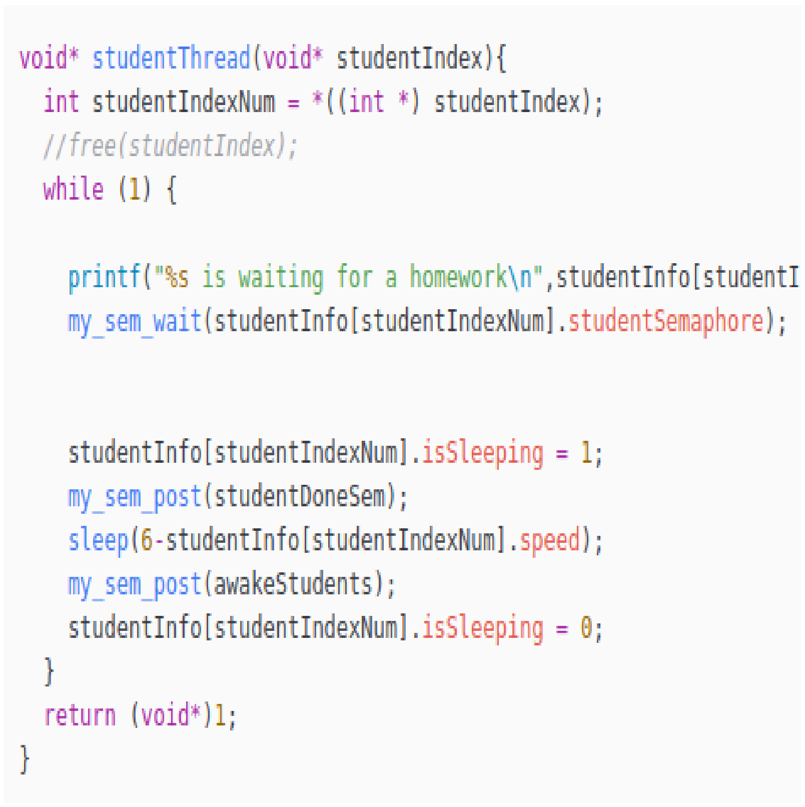
Figure 1.3: Implementation of the Main Thread

3.3 Student Thread Design and Implementation

Student threads are designed to do homeworks when it is assigned to them.According to the features they have,for each homework there are different suitable students.(Speed,Cost,Quality).

All students should wait until a homework assigned to them.This is provided with semaphore for each student thread.After a student thread is triggered by main thread, student takes money and in order to do the homework they wait for specified amount of time.

NOTE: This figure is not the latest version.It only used to explain the concept(Signal handling part is not shown yet for the sake of simplicity).



```
void* studentThread(void* studentIndex){
  int studentIndexNum = *((int *) studentIndex);
  //free(studentIndex);
  while (1) {

    printf("%s is waiting for a homework\n",studentInfo[studentI
    my_sem_wait(studentInfo[studentIndexNum].studentSemaphore);


    studentInfo[studentIndexNum].isSleeping = 1;
    my_sem_post(studentDoneSem);
    sleep(6-studentInfo[studentIndexNum].speed);
    my_sem_post(awakeStudents);
    studentInfo[studentIndexNum].isSleeping = 0;
  }
  return (void*)1;
}
```

Figure 1.5: Implementation of Student Threads

4 Semaphores

In order for to provide synchronization between processes, 5 POSIX Semaphores are used in this project:

• MUTEX

Mutex is used to lock the critical region in order to prevent existence of multiple threads in the critical region simultaneously.

• EMPTY

Empty Semaphore is used to prevent the producer threads to enter the critical region if there is not enough room in the queue.In this case, producer thread should wait until main thread consume the homeworks.

• AWAKESTUDENTS

This semaphore is used to count the awake students in order to let main thread sleep while all of the students are busy with a homework.After students finish sleeping,they will post this semaphore.

• STUDENTDONE

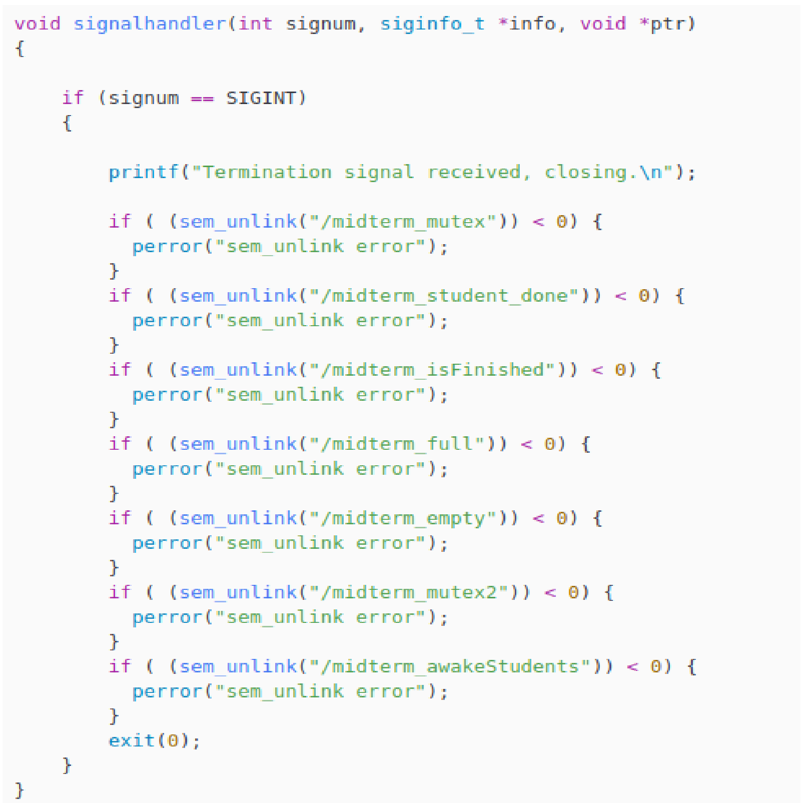
CitizenDone semaphore is used to block main thread to trigger other students while current student is in critical region. The main thread who triggered the student waits for this semaphore until that student is finished with the critical region(doesn't wait for sleeping). After a student finished with critical region, student post that semaphore to let main thread leave the critical region nicely.

• STUDENT SEMAPHORES

Every student has their own semaphore in order to provide synchronization between main thread and students.While students are waiting for their semaphore values to be posted,main thread selects the best suitable student for a homework and post corresponding semaphore in order to trigger that student to do homework.

5 Signal Handler

When the user press CTRL-C while program is still running , in order to terminate the program gracefully all controls are added to thread functions.
If user presses CTRL-C then it triggers signal handler function which deallocate spaces nicely and delete all semaphores that is created at the beginning of the program.



```
void signalhandler(int signum, siginfo_t *info, void *ptr)
{

    if (signum == SIGINT)
    {

        printf("Termination signal received, closing.\n");

        if ( (sem_unlink("/midterm_mutex")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_student_done")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_isFinished")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_full")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_empty")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_mutex2")) < 0) {
          perror("sem_unlink error");
        }
        if ( (sem_unlink("/midterm_awakeStudents")) < 0) {
          perror("sem_unlink error");
        }
        exit(0);
    }
}
```

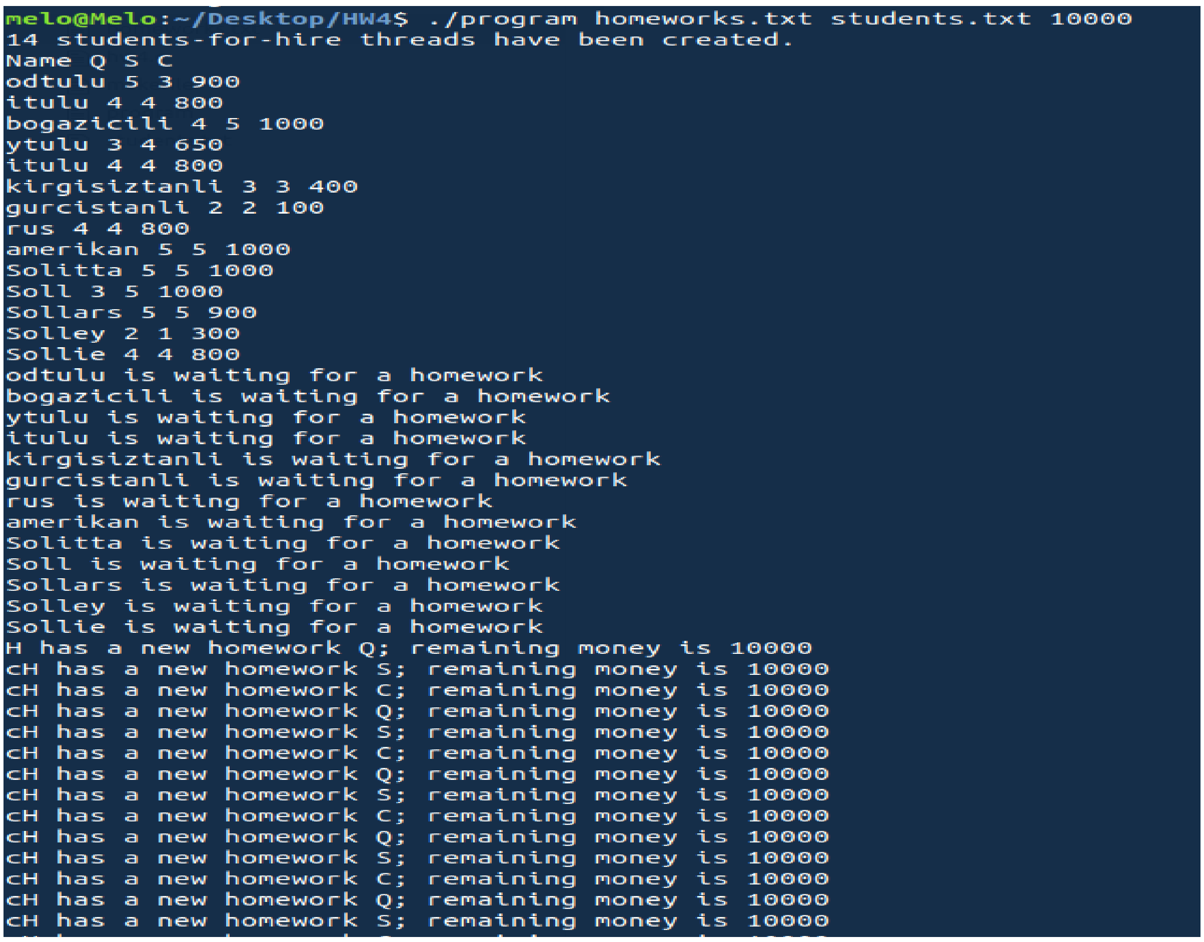
Figure 1.6: Signal Handler

6 Running Results

6.1 Test Case 1

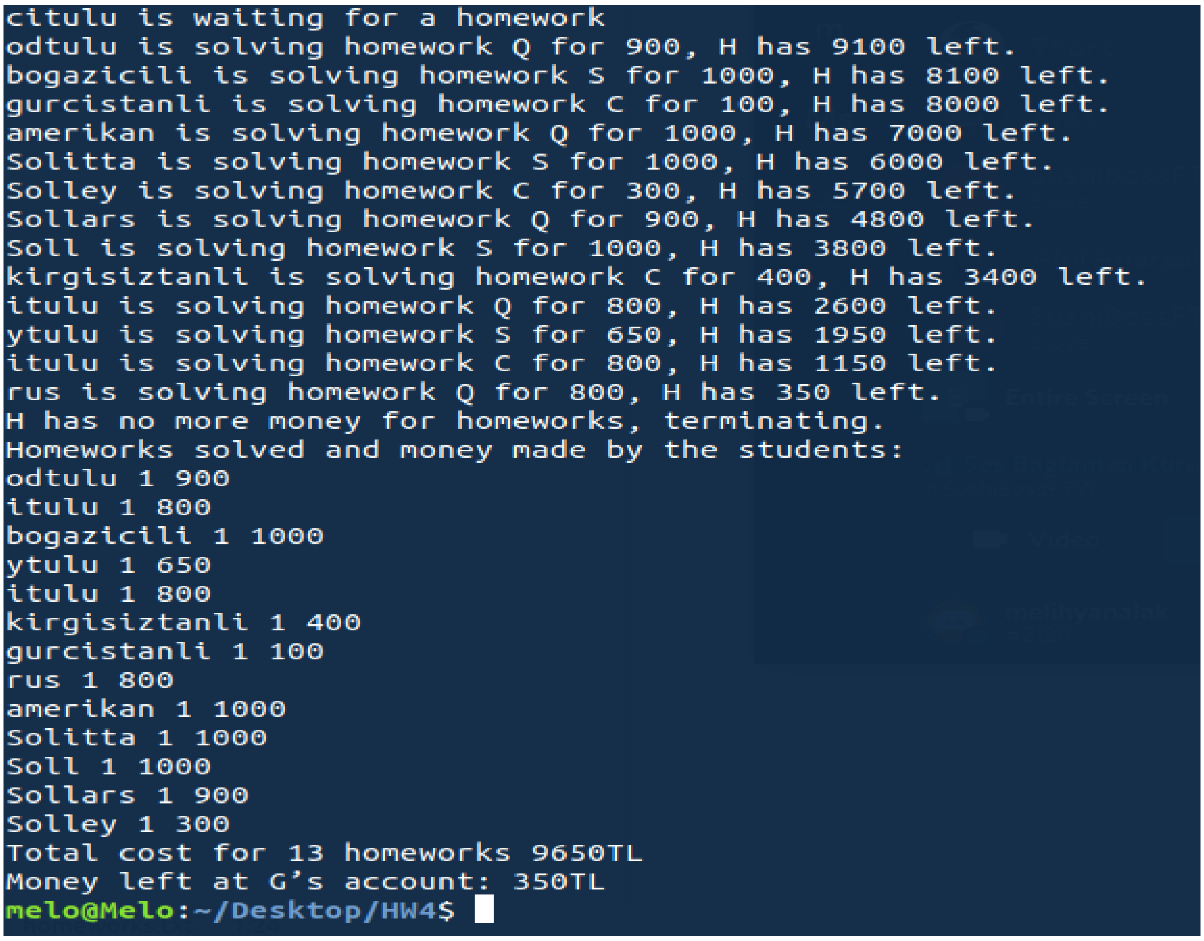
The program is tested with many different input files and parameters.
In first test case input file is like this:

QSCQSCQSCQSCQSCQSCQSCQS
Then the output of the program is below:



```
melo@Melo:~/Desktop/HW4$ ./program homeworks.txt students.txt 10000
14 students-for-hire threads have been created.
Name Q S C
odtulu 5 3 900
itulu 4 4 800
bogazicili 4 5 1000
ytulu 3 4 650
itulu 4 4 800
kirgisiztanli 3 3 400
gurcistanli 2 2 100
rus 4 4 800
amerikan 5 5 1000
Solitta 5 5 1000
Soll 3 5 1000
Sollars 5 5 900
Solley 2 1 300
Sollie 4 4 800
odtulu is waiting for a homework
bogazicili is waiting for a homework
ytulu is waiting for a homework
itulu is waiting for a homework
kirgisiztanli is waiting for a homework
gurcistanli is waiting for a homework
rus is waiting for a homework
amerikan is waiting for a homework
Solitta is waiting for a homework
Soll is waiting for a homework
Sollars is waiting for a homework
Solley is waiting for a homework
Sollie is waiting for a homework
H has a new homework Q; remaining money is 10000
cH has a new homework S; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework Q; remaining money is 10000
cH has a new homework S; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework Q; remaining money is 10000
cH has a new homework S; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework Q; remaining money is 10000
cH has a new homework S; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework Q; remaining money is 10000
cH has a new homework S; remaining money is 10000
```

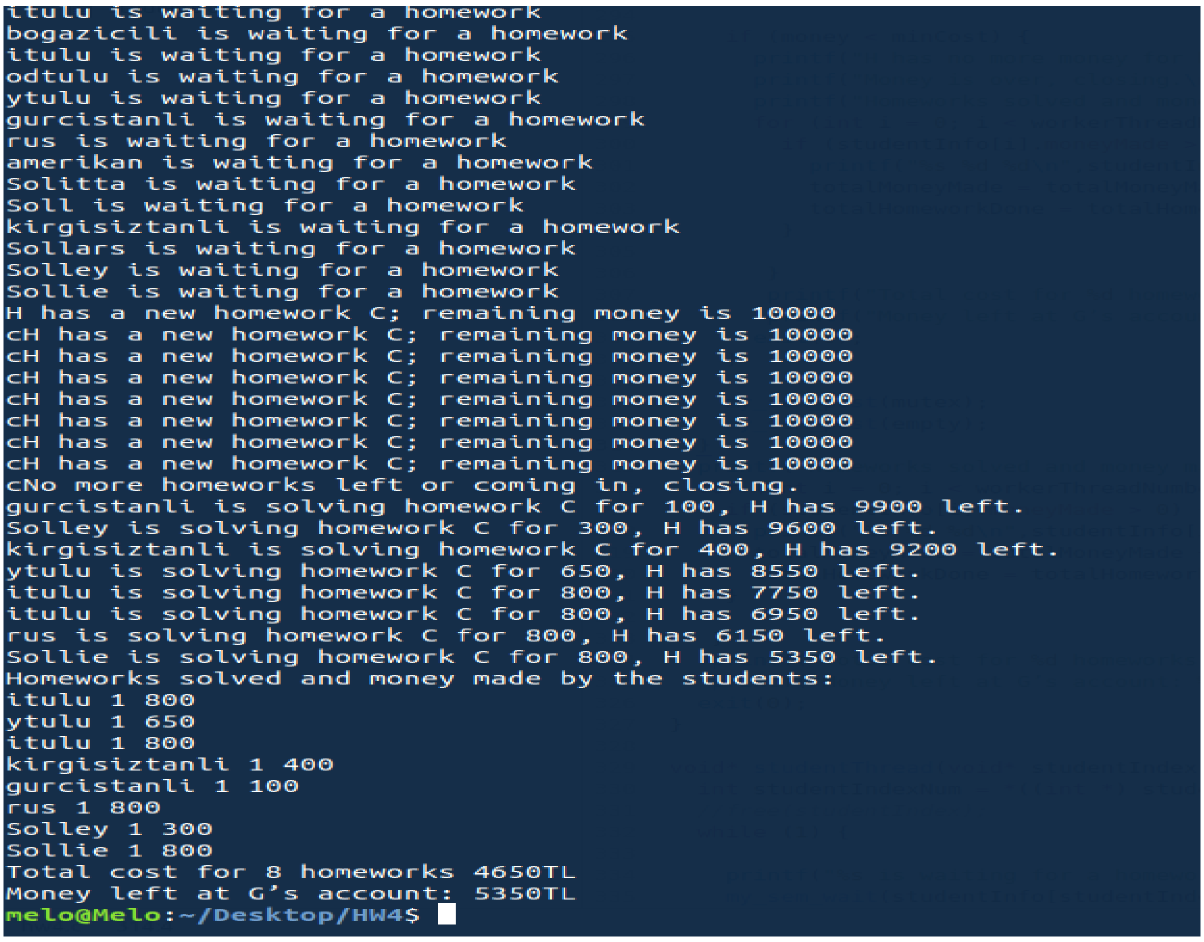
Figure 1.7: Example Output



```
citulu is waiting for a homework
odtulu is solving homework Q for 900, H has 9100 left.
bogazicili is solving homework S for 1000, H has 8100 left.
gurcistanli is solving homework C for 100, H has 8000 left.
amerikan is solving homework Q for 1000, H has 7000 left.
Solitta is solving homework S for 1000, H has 6000 left.
Solley is solving homework C for 300, H has 5700 left.
Sollars is solving homework Q for 900, H has 4800 left.
Soll is solving homework S for 1000, H has 3800 left.
kirgisiztanli is solving homework C for 400, H has 3400 left.
itulu is solving homework Q for 800, H has 2600 left.
ytulu is solving homework S for 650, H has 1950 left.
itulu is solving homework C for 800, H has 1150 left.
rus is solving homework Q for 800, H has 350 left.
H has no more money for homeworks, terminating.
Homeworks solved and money made by the students:
odtulu 1 900
itulu 1 800
bogazicili 1 1000
ytulu 1 650
itulu 1 800
kirgisiztanli 1 400
gurcistanli 1 100
rus 1 800
amerikan 1 1000
Solitta 1 1000
Soll 1 1000
Sollars 1 900
Solley 1 300
Total cost for 13 homeworks 9650TL
Money left at G's account: 350TL
melo@Melo:~/Desktop/HW4$
```

Figure 1.8: Example Output

6.2 Test Case 2



```
itulu is waiting for a homework
bogazicili is waiting for a homework
itulu is waiting for a homework
odtulu is waiting for a homework
ytulu is waiting for a homework
gurcistanli is waiting for a homework
rus is waiting for a homework
amerikan is waiting for a homework
Solitta is waiting for a homework
Soll is waiting for a homework
kirgisiztanli is waiting for a homework
Sollars is waiting for a homework
Solley is waiting for a homework
Sollie is waiting for a homework
H has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cH has a new homework C; remaining money is 10000
cNo more homeworks left or coming in, closing.
gurcistanli is solving homework C for 100, H has 9900 left.
Solley is solving homework C for 300, H has 9600 left.
kirgisiztanli is solving homework C for 400, H has 9200 left.
ytulu is solving homework C for 650, H has 8550 left.
itulu is solving homework C for 800, H has 7750 left.
itulu is solving homework C for 800, H has 6950 left.
rus is solving homework C for 800, H has 6150 left.
Sollie is solving homework C for 800, H has 5350 left.
Homeworks solved and money made by the students:
itulu 1 800
ytulu 1 650
itulu 1 800
kirgisiztanli 1 400
gurcistanli 1 100
rus 1 800
Solley 1 300
Sollie 1 800
Total cost for 8 homeworks 4650TL
Money left at G's account: 5350TL
melo@Melo:~/Desktop/HW4$
```

Figure 1.9: Example Output 2