IAX0584 Programming II
Files and structures
Homework no.1

Ivan Symotiuk
223686MVEB

Supervisor: Associate Professor Vladimir Viies

Copyright declaration

I have prepared this work independently. All the work of other authors used in the preparation of the work, important points of view, data from literary sources, and the seller are cited.

Ivan Symotiuk

Contents

Setting up a task

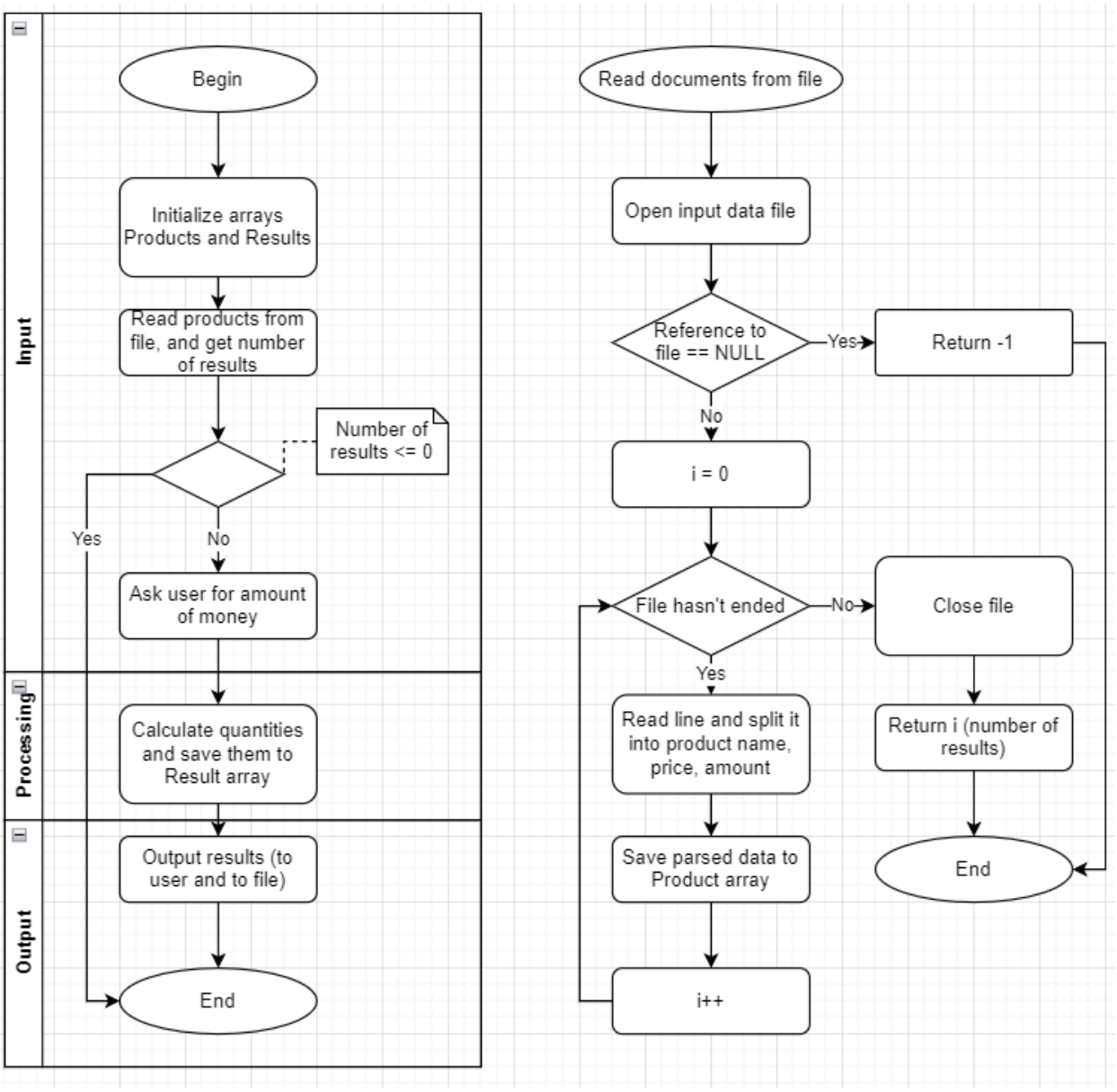
The task is to create a program that takes data about products (name, price, and available amount), asks the user for available money, and prints out what he can buy. The data is stored in a text file “F1.txt”. For example:

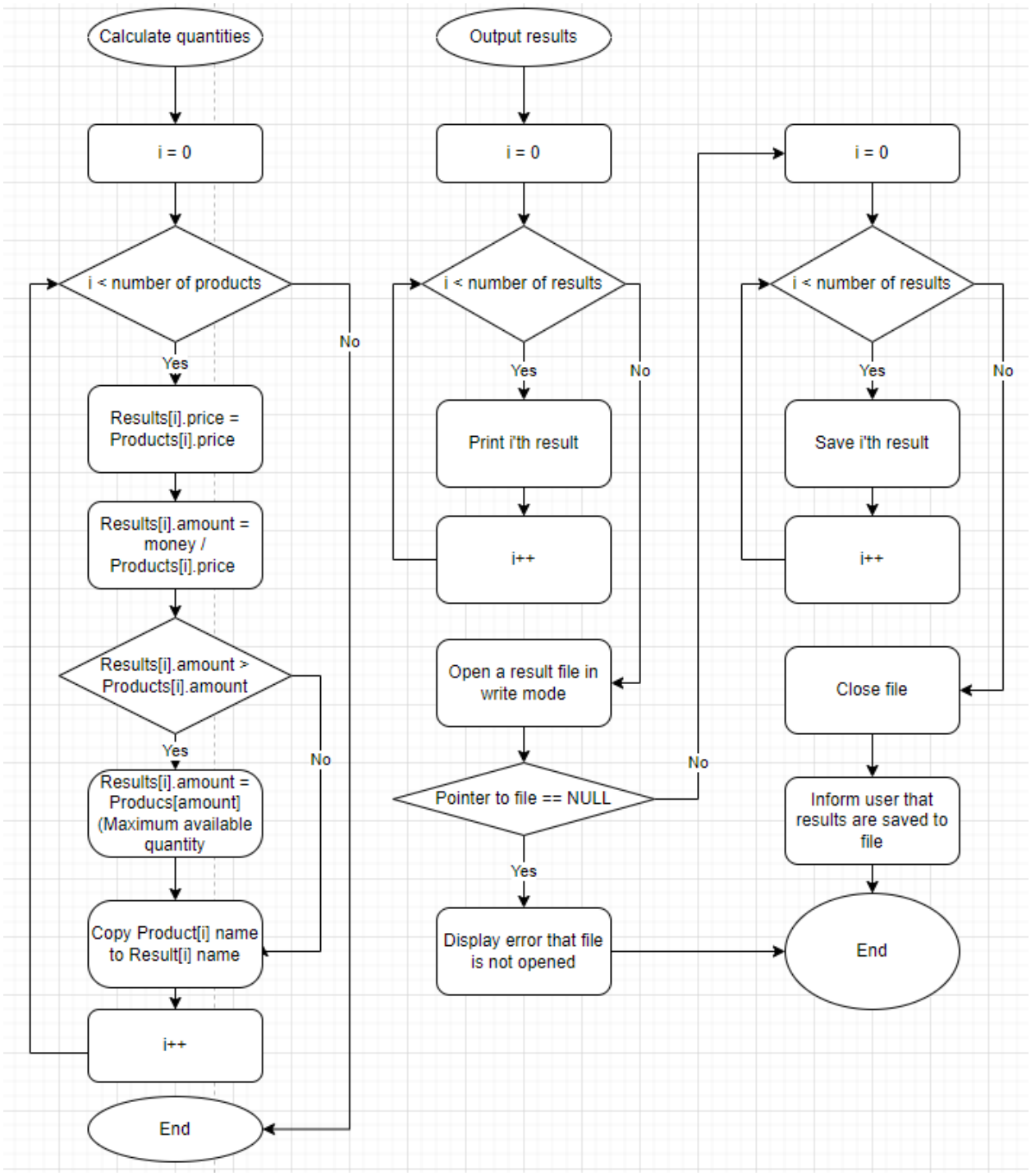
```
Sugar      1.19 1
Juice      2.19 1
Rice       0.19 4
Beans     40.19 3
Oil        2.19 1
Eggs       2.19 2
Bread      2.19 1
Potato     0.19 2
Onion      0.19 3
```

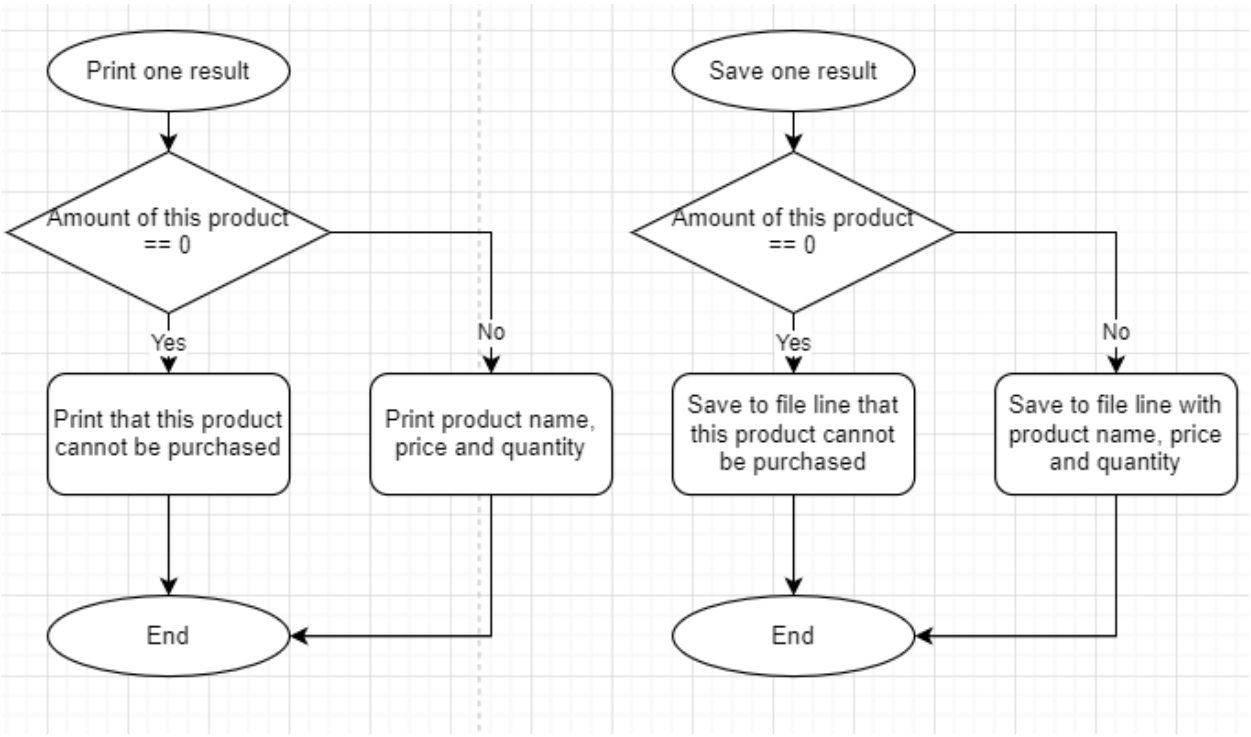
The result is a list of the product, their prices, and the number of items the user can buy. If the user cannot purchase this type of product, the program should output a warning message. For example:

```
Sugar    - 1.19 x 1
Juice    - No products can be purchased
Rice     - 0.19 x 4
Beans    - No products can be purchased
Oil      - No products can be purchased
Eggs     - No products can be purchased
Bread    - No products can be purchased
Potato   - 0.19 x 2
Onion    - 0.19 x 3
```

Algorithm


Input
Processing
Output
Begin
Initialize arrays Products and Results
Read products from file, and get number of results
Number of results <= 0
Yes
No
Ask user for amount of money
Calculate quantities and save them to Result array
Output results (to user and to file)
End
Read documents from file
Open input data file
Reference to file == NULL
Yes
Return -1
No
i = 0
File hasn't ended
No
Close file
Yes
Read line and split it into product name, price, amount
Return i (number of results)
Save parsed data to Product array
End
i++


Calculate quantities
i = 0
i < number of products
Yes
No
Results[i].price = Products[i].price
Results[i].amount = money / Products[i].price
Results[i].amount > Products[i].amount
Yes
No
Results[i].amount = Producs[amount] (Maximum available quantity
Copy Product[i] name to Result[i] name
i++
End
Output results
i = 0
i < number of results
Yes
No
Print i'th result
i++
Open a result file in write mode
Pointer to file == NULL
No
Yes
Display error that file is not opened
i = 0
i < number of results
Yes
No
Save i'th result
i++
Close file
Inform user that results are saved to file
End


Print one result
Amount of this product == 0
Yes
No
Print that this product cannot be purchased
Print product name, price and quantity
End
Save one result
Amount of this product == 0
Yes
No
Save to file line that this product cannot be purchased
Save to file line with product name, price and quantity
End

Folder structure

The code of the program is divided into files. The root of the project will look like this:

/project
- main.c
- F1.txt
- F2.txt
- functions.h
- functions.c

The *main.c* file is located at the root of the project and is the entry point of the program. *F1.txt* and *F2.txt* are input and output files respectively.

File *functions.c* contains all necessary declarations of functions and *functions.h* contains their prototypes and structure declaration

Source code

In the next sections, there is a source code of the program

main.c

```c
#include "functions.h"

int main()
{
    Product p[MAX_PRODUCT_NUMBER];
    Result  r[MAX_PRODUCT_NUMBER];

    int numProducts = getProducts(INPUT_FILE, p);

    if(numProducts <= 0) {
        printf("No products found in file %s");
        return -1;
    }

    float money = getMoney("Enter money: ");

    calculateResults(money, numProducts, p, r);
    outputResults(OUTPUT_FILE, numProducts, r);

    return 0;
}
```

functions.h

```c
#ifndef FUNCTIONS_H
#define FUNCTIONS_H

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#define BUFFER_SIZE 100
```

```c
#define MAX_NAME_SIZE 40
#define MAX_PRODUCT_NUMBER 15

#define INPUT_FILE "F1.txt"
#define OUTPUT_FILE "F2.txt"

typedef struct Product {
    char name[MAX_NAME_SIZE];
    float price;
    int amount;
} Product;

typedef struct Result {
    char name[MAX_NAME_SIZE];
    float price;
    int amount;
} Result;

int getProducts(char *, Product[]);
float getMoney(char *);
void calculateResults(float, int, Product [], Result []);
void outputResults(char *, int, Result []);

#endif // FUNCTIONS_H
```

functions.c

```c
#include "functions.h"

Product parseLine(char *);
void printResult(Result);
void saveResult(Result, FILE *);


int getProducts(char *fileName, Product p[]) {
    FILE *fp;
    fp = fopen(fileName, "r");
```

```
    if (fp == NULL)
    {
        printf("Cannot open file %s", fileName);
        return -1;
    }

    int i = 0;
    while (!feof(fp)) {
        char lineBuffer[100];
        fgets(lineBuffer, sizeof(lineBuffer), fp); // read one line to
buffer
        p[i] = parseLine(lineBuffer);
        i++;
    }
    fclose(fp);
    return i;
}

Product parseLine(char *line) {
    Product p;
    sscanf(line, "%s %f %d", p.name, &p.price, &p.amount);
    return p;
}

float getMoney(char *message) {
    float money;
    do {
        printf("%s", message);
        scanf("%f", &money);
    } while (money <= 0);
    return money;
}

void calculateResults(float money, int numProducts, Product p[], Result
r[]) {
    int i = 0;
    for(i = 0; i < numProducts; i++) {
        r[i].price = p[i].price;
        r[i].amount = money / p[i].price;
        if(r[i].amount > p[i].amount) {
```

```
            r[i].amount = p[i].amount;
        }
        strcpy(r[i].name, p[i].name);
    }
}

void outputResults(char * filename, int numResults,  Result r[]) {
    for (int i = 0; i < numResults; i++) {
        printResult(r[i]);
    }

    FILE *fp;
    fp = fopen(filename, "w");
    if (fp == NULL) {
        printf("Failed saving the results to %s", filename);
        return;
    }

    for (int i = 0; i < numResults; i++) {
        saveResult(r[i], fp);
    }

    fclose(fp);
    printf("Results saved to %s", filename);
}

void printResult(Result r) {
    if(r.amount == 0) {
        printf("%-8s - No products can be purchased \n", r.name);
    } else {
        printf("%-8s - %.2f x %d\n", r.name, r.price, r.amount);
    }

}

void saveResult(Result t, FILE *fp) {
    if(t.amount == 0) {
        fprintf(fp, "%-8s - No products can be purchased \n", t.name);
    } else {
        fprintf(fp, "%-8s - %.2f x %d\n", t.name, t.price, t.amount);
```

```
    }
}
```

Code Explanation

At first, the program declares arrays: Products and Results of corresponding structures. Here input and output data will be stored respectively.

The program takes data from file F1.txt as input. At first, it checks if the file is successfully opened, reads it line by line, and parses each line to fill a *Product* structure. This data is stored in an array of *Products*

After that the program checks for errors (then a number of products are -1) or the absence of data (then the number will be 0). If there is an error the program displays a message and finishes execution. In another case, it proceeds.

After that, the program asks users for available money and calculates the result. For each product, it copies its name and price to the *Result* array. Then it calculates how many full items the customer can buy. If he can buy more, then the program limits this value to the number available in stock

At the end program prints the *Result* array and saves it to *F2.txt*