ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА
Факултет по изчислителна техника и автоматизация
Катедра „КНТ“

СЕМЕСТРИАЛНА ДОМАШНА РАБОТА

по дисциплината „**Базово програмиране**”

на тема: „ **Информационна система Сервиз за техника**”

Вариант: **259**

Изготвил:
Николай Миленов Симеонов

Проверил:
проф. М. Карова
гл.ас. др. Е. Ракитина

Специалност:
Изкуствен интелект
Група: 1А

2025

Факултетен номер: 24621905

Съдържание

Задание на проекта

Информационна система Сервиз за техника

Да се напише компютърна програма, реализираща информационна система за сервиз за

техника. Програмата съхранява информация за извършените сервизни поръчки в рамките на

даден месец (номер на поръчката, ден от месеца, клиент, тип устройство (телефон, лаптоп,

телевизор и др.), сериен номер на устройството, проблем според клиента (до 50 символа), име

на сервизния техник, извършен ремонт (до 50 символа), цена, дни престой в сервиза, статус:

приета/отказана/завършена). Предполагаемите проблеми и извършените ремонти да бъдат

предварително декларирани в отделни списъци с максимален брой по 10 за всеки списък.

Максималния брой сервизни поръчки е 100.

Базова задача

A. Меню за избор на функциите от програмата

Функции от програмата са:

B. Добавяне на нова сервизна поръчка:

a. Добавяне на няколко поръчки, като не трябва да се превишава максималният брой

(100).

Пример: Добавяне на списък от поръчки. Въвежда се цяло число n и след него n на

брой поръчки. n не може да надвишава свободните елементи в списъка с поръчки.

*При добавяне на поръчка се въвежда част от информацията (номер на

поръчка, вид устройство, номер, и т.н.). Име на сервизен техник, извършен

ремонт и т.н. се актуализират след извършения ремонт!

C. Извеждане на всички устройства на екрана:

a. Извеждане на всички устройства, чрез подходящо форматиране в таблица

D. Търсене и извеждане на екрана:

a. Търсене и извеждане на устройства по вид

b. Търсене и извеждане на устройства по статус

E. Сортиране и извеждане на поръчки на екран:

a. Сортиране на поръчките по дата на приемане

F. Работа с външен файл(двоичен)

a. Извеждане на масива с поръчки във файл.

b. Въвеждане на масива с поръчки от файл.

Допълнение първо (+ базова задача)

G. Да се създаде подменю към основното с нови функции за:

a. Отделяне на завършените поръчки на определен техник и сортирани по ден на

приемане.

b. Отделяне на поръчките с определен проблем за дадено устройство(въвежда се от

потребителя) и сортиране в намаляващ ред по тип на устройството.

Допълнение второ (+ базова задача)

H. Въвеждане/Актуализация на данни за сервизна поръчка:

a. Добаване в структурата на поле вид на поръчката (нормална – за 10 дена, бърза

за 5 дена, експресна за 2 дена)

b. Въвеждане/Актулизацията на данни (за извършен ремонт) става по въведен номер

на сервизната поръчйа. Ако поръчката присъства в списъка(масива):

a. Проверява се статуса й:

Приета поръчка – се актуализира информацията за извършеният ремонт и се пресмята цената

(Изчисляване на цена: всеки тип ремонт има цена (в лв.) + процент от цената на ремонта за

вида поръчка (експресна - 50%, бърза - 20%, нормална - 0% от цената на ремонт).

Отказана/завършена – НЕ може да се актуализира информация

b. Ако не присъства в масива, се извежда пояснително съобщение на екрана.

I. Смяна на статус на поръчка: Въвежда се номерът на поръчката (ако я има се въвежда и

новият статус). Ако статусът е приета се сменя с новият. Ако статусът е

отказана/завършена НЕ СЕ допуска промяна!

Допълнение трето (+ базова задача)

J. Данните в програмата да се попълват автоматично от файл при стартиране и да се

записват автоматично във файл при затваряне на програмата.

I. Анализ на решението

1. Структура за данните в програмата

Структура	Обяснение	Примерени стойности

enum RequestType {
NORMAL,
FAST,
EXPRESS
};
 	
enum RequestStatus {
PENDING,
REJECTED,
COMPLETED
};
 	
struct Request {
int requestNumber = 0;
char date[MAX_STRING_LENGTH]{ '0' };
char clientName[MAX_STRING_LENGTH]{ '0' };
char deviceType[MAX_STRING_LENGTH]{ '0' };
char serialNumber[MAX_STRING_LENGTH]{ '0' };
char issueDescription[MAX_STRING_LENGTH] { '0' };
char technicianName[MAX_STRING_LENGTH] { "PENDING" };
char repairDetails[MAX_STRING_LENGTH] { "PENDING" };
float repairCost = 0;
 	
RequestStatus status;
RequestType type;
};
	– requestNumber - int променлива, която съхранява поредния номер на заявката
	– date - char[] променлива която съхранява датата на подаване на заявката въведена от клиента
	– clientName - char[] променлива, която съхранява името на клиента
	– deviceType - char[] променлива, която съхранява типа на устройството с което е свързана заявката
	– serialNumber - char[] променлива, която съхранява серийния номер на у-вото
	– issueDescription - char[] променлива, която съдържа описание на проблема с у-вото
	– technicianName - char[] променлива, която съхранява името на техникът, работил по заявката
	– repairDetails - char[] променлива, която съдържа информация за извършената поправка
	– repairCost - float променлива, която съдържа цената, която трябва да се заплати.
		3
		
		15-12-2024
		Книга1
		
		Иван Георгиев
		
		Phone
		
		2ndi282diuhsud
		
		Overheating issue
		
		Григов Петров
		
		Сменена батерия
		
		250
		
		PENDING(0)
		
		NORMAL(0)

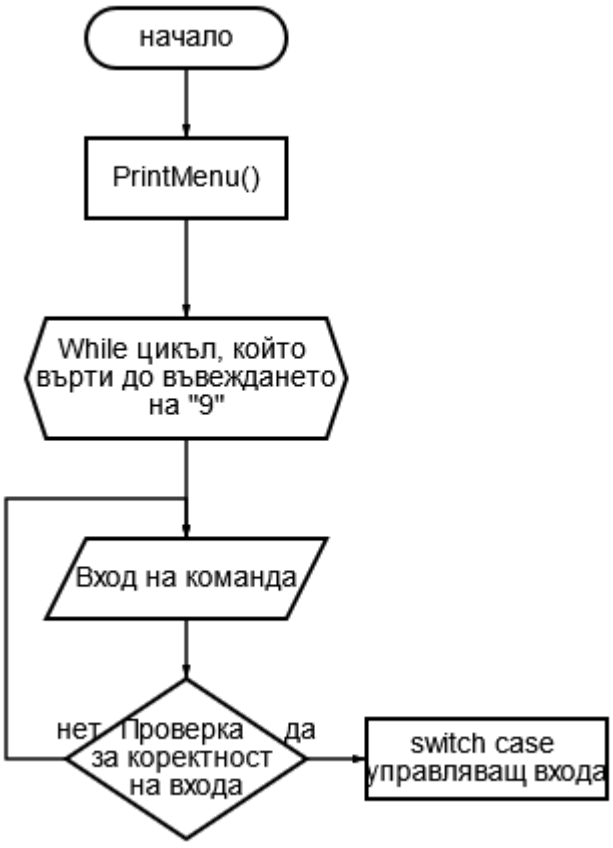
	– Status - enum държащ всички възможни статуси на заявка – Type - enum държащ всички възможни видове заявка	

2. Реализация на условие A

2.1. Анализ на алгоритъма

След зареждане на програмата менюто автоматично бива изведено още на първият ред от main() функцията.

2.2. Блок схема на алгоритъма



2.3. Функция, с която е реализиран алгоритъма

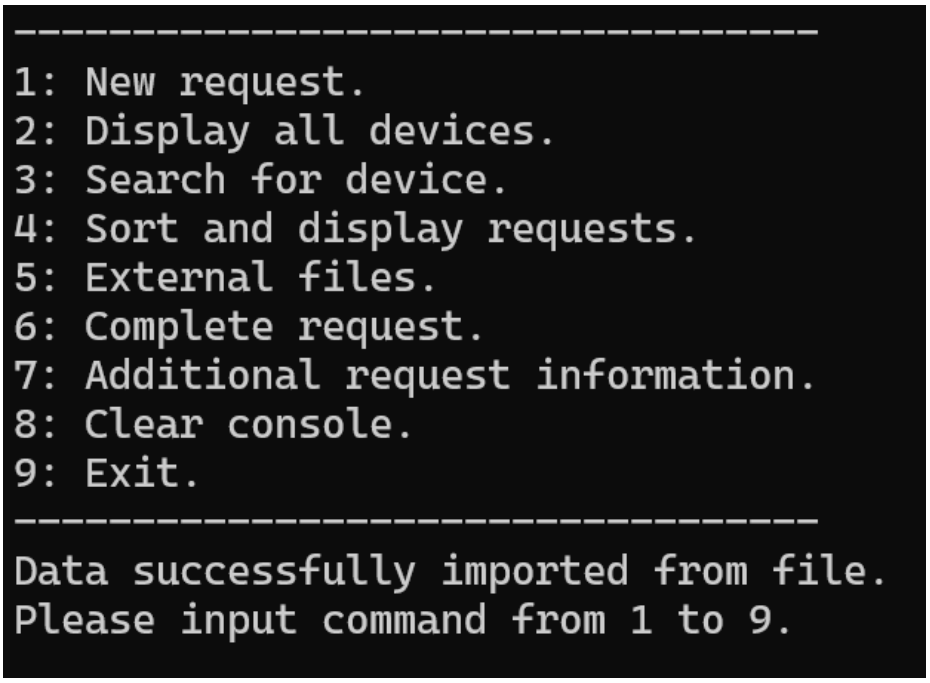
```
void PrintMenu()
{
    cout << "--------------------------------" << endl;
    cout << "1: New request." << endl;
    cout << "2: Display all devices." << endl;
    cout << "3: Search for device." << endl;
```

```
    cout << "4: Sort and display requests." << endl;
    cout << "5: External files." << endl;
    cout << "6: Complete request." << endl;
    cout << "7: Additional request information." << endl;
    cout << "8: Clear console." << endl;
    cout << "9: Exit." << endl;
    cout << "-----------------------------------" << endl;
}

int main()
{
    PrintMenu();
    const int MAX_REQUESTS = 100;
    Request reqArray[MAX_REQUESTS];
    int inputtedRequests = 0;
    ImportAndExportBinaryFileAutomatically(reqArray, inputtedRequests, false);
    int command;
    do
    {
        command = InputCommand(command);

        if (command < 1 || command > 9 || cin.fail())
        {
            cout << "Wrong command!" << endl;
        }
        else
        {
            switch (command)
            {
            case 1: InputRequest(reqArray, inputtedRequests); break;
            case 2: DisplayDevices(reqArray, inputtedRequests); break;
            case 3: SearchForRequest(reqArray, inputtedRequests); break;
            case 4: SortByDate(reqArray, inputtedRequests); break;
            case 5: ImportAndExportBinaryFile(reqArray, inputtedRequests); break;
            case 6: CompleteRequest(reqArray, inputtedRequests); break;
            case 7: AdditionalRequestInformation(reqArray, inputtedRequests); break;
            case 8: system("cls"); PrintMenu(); break;
            }
        }
    } while (command != 9);
    ImportAndExportBinaryFileAutomatically(reqArray, inputtedRequests, true);
}
```

2.3.1. Входни данни на функцията

Въвеждат се числата от 1 до 9 с които се избира следващото действие, което ще извърши програмата.

2.3.2. Изходни данни на функцията или данни, които се извеждат

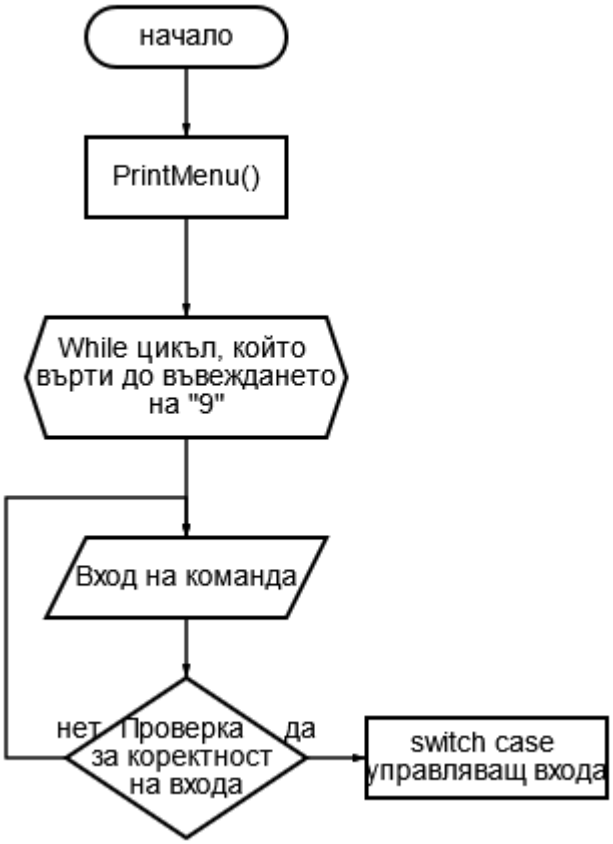


3. Реализация на условие B

3.1. Анализ на алгоритъма, който трябва да се реализира

След избиране на функция две от конзолата потребителят ще бъде попитан колко заявки желае да въведе, след което ще бъде питан да въведе или избере стойност за всяка променлива от структурата, която трябва да бъде попълнена при създаването и. Когато приключи с първата, ще бъде уведомен и ще започне въвеждане на следващата.

3.2. Блок схема на алгоритъма



3.3. Функция, с която е реализиран алгоритъма

```
void InputRequest(Request reqArray[], int& inputtedRequests)
{
    int n = 0;

    bool correctInput = false;
    while (!correctInput)
    {
        cout << "Please enter the number of requests you want to input:";
        cin >> n;
        if (cin.fail() || n <= 0 || n > 100)
        {
            cin.clear();
            cin.ignore();
            cout << "Invalid input. Please enter a positive number less than or equal to 100."
<< endl;
        }
        else
        {
            correctInput = true;
        }
    }
```

```
    for (int i = 0; i < n; i++)
    {
        cin.ignore();
        cout << "";
        reqArray[inputtedRequests].requestNumber = inputtedRequests + 1;
        cout << "Request Number is: " << reqArray[inputtedRequests].requestNumber <<
endl;

        cout << "Enter Date (DD-MM-YYYY): ";
        cin.getline(reqArray[inputtedRequests].date, MAX_STRING_LENGTH);

        cout << "Enter Client Name: ";
        cin.getline(reqArray[inputtedRequests].clientName, MAX_STRING_LENGTH);

        for (int j = 0; j < 10; j++)
        {
            cout << j + 1 << ": " << deviceTypes[j] << endl;
        }
        int deviceTypeIndex;
        cout << "Select Device Type: ";
        cin >> deviceTypeIndex;
        deviceTypeIndex--;
        while (deviceTypeIndex < 0 || deviceTypeIndex >= 10 || cin.fail())
        {
            cout << "Invalid selection. Please select a valid device type index: ";
            cin >> deviceTypeIndex;
        }
        strcpy_s(reqArray[inputtedRequests].deviceType, deviceTypes[deviceTypeIndex]);

        cout << "Enter Serial Number: ";
        cin.ignore();
        cin.getline(reqArray[inputtedRequests].serialNumber, MAX_STRING_LENGTH);

        for (int j = 0; j < 10; j++)
        {
            cout << j + 1 << ": " << issueDescriptions[j] << endl;
        }
        int issueDescriptionIndex;
        cout << "Select Issue Description: ";
        cin >> issueDescriptionIndex;
        issueDescriptionIndex--;
        while (issueDescriptionIndex < 0 || issueDescriptionIndex >= 10 || cin.fail())
        {
            cout << "Invalid selection. Please select a valid issue description index: ";
            cin >> issueDescriptionIndex;
        }
        strcpy_s(reqArray[inputtedRequests].issueDescription,
issueDescriptions[issueDescriptionIndex]);

        cout << "Technician Name is pending. " << endl;
```

```
        cout << "Repair Details are pending. " << endl;
        cout << "Repair Cost is pending. " << endl;

        int typeInput;
        bool fine = false;
        do
        {
            cout << "Enter Status (0 for NORMAL, 1 for FAST, 2 for EXPRESS): ";
            cin.ignore();

            cin >> typeInput;
            if (typeInput > 2 || typeInput < 0)
            {
                cout << "Wrong input! Try Again: (0 for NORMAL, 1 for FAST, 2 for
EXPRESS): ";
            }
            else
            {
                fine = true;
            }
        } while (fine == false);
        cin.ignore();

        reqArray[inputtedRequests].type = static_cast<RequestType>(typeInput);
        reqArray[inputtedRequests].status = static_cast<RequestStatus>(0);
        cout << "\ nRequest data successfully entered!\n\n" << "---------------------------------"
<< endl;
        inputtedRequests += 1;
    }
}
```

3.3.1. Входни данни на функцията

Цяло число n (брой заявки).

За всяка поръчка: номер на поръчка, вид устройство, статус и друга основна информация.

3.3.2. Изходни данни на функцията или данни, които се извеждат

Извежда се съобщение „Request data successfully entered!“

4. Реализация на условие C

4.1. Анализ на алгоритъма, който трябва да се реализира

Алгоритъмът принтира първо горната част описваща полетата на таблицата, след което ред по ред изписва всяка заявка.

4.2. Функция, с която е реализиран алгоритъма

```
void DisplayDevices(Request const reqArray[], int const requestsToFind)
{
```

```
    cout << left << setw(9) << "Request"
         << setw(12) << "Date"
         << setw(20) << "Client Name"
         << setw(15) << "Device Type"
         << setw(15) << "Serial No"
         << setw(25) << "Issue Description"
         << setw(20) << "Technician Name"
         << setw(25) << "Repair Details"
         << setw(4) << "Cost"
         << setw(10) << "Status"
         << setw(10) << "Type"
         << endl;

    cout << string(162, '-') << endl;

    for (int i = 0; i < requestsToFind; i++)
    {
        cout << left << setw(9) << reqArray[i].requestNumber
             << setw(12) << reqArray[i].date
             << setw(20) << reqArray[i].clientName
             << setw(15) << reqArray[i].deviceType
             << setw(15) << reqArray[i].serialNumber
             << setw(25) << reqArray[i].issueDescription
             << setw(20) << reqArray[i].technicianName
             << setw(25) << reqArray[i].repairDetails
             << setw(4) << reqArray[i].repairCost
             << setw(10) << (reqArray[i].status == PENDING ? "PENDING" :
(reqArray[i].status == REJECTED ? "REJECTED" : "COMPLETED"))
             << setw(10) << (reqArray[i].type == NORMAL ? "NORMAL" : (reqArray[i].type ==
FAST ? "FAST" : "EXPRESS"))
             << endl;
    }
}
```

4.2.1. Входни данни на функцията

Няма допълнителни входни данни – използва текущия списък с поръчки.

4.2.1. Изходни данни на функцията или данни, които се извеждат

```
Request  Date        Client Name        Device Type    Serial No      Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob         Phone          lej928f        Battery issue           Robert Ratakowski   ew battery installed    150 COMPLETED NORMAL
2        21-12-2023  Eduard Ivanov      Tablet         asjdo9q        Overheating             PENDING             PENDING                 0   REJECTED  EXPRESS
3        12-09-2019  Mihaela Filipova   Laptop         ijhguig        Software crash          Jordan Philipow     einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski   Phone          khugug         No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
```

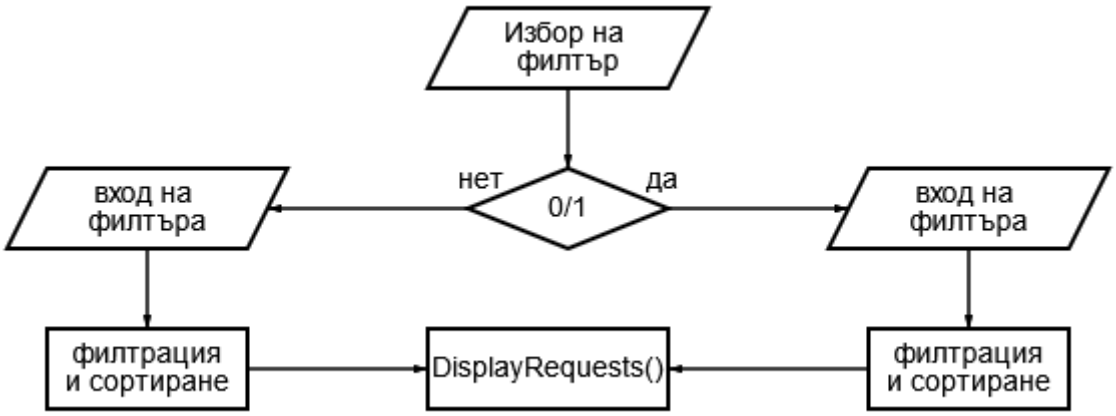
5. Реализация на условие D

5.1. Анализ на алгоритъма, който трябва да се реализира

Потребителят избира по какъв критерий ще филтрира и търси, след което ако търси по тип устройство въвежда какво е то, а ако търси по статус въвежда него.

След това на функцията за извеждане на заявки се подават нови масиви съставени от филтрираната информация.

5.2. Блок схема на алгоритъма



5.3. Функция с която е реализиран алгоритъма

```
void SearchForRequest(Request reqArray[], int const inputtedRequests)
{
    cout << "Search for request by device type(0) or request status(1)." << endl;
    cin.ignore();

    int searchType;
    bool fine = false;
    const int maxRequests = 100;
    int foundCount = 0;
    Request foundRequests[maxRequests];

    do
    {
        cin >> searchType;
        if (searchType < 0 || searchType>1)
        {
            cout << "Wrong input! Try Again: (0 for device type, 1 for request status): ";
            cin.ignore();
        }
        else fine = true;
    } while (fine == false);

    if (searchType == 0)
    {
        char searchString[MAX_STRING_LENGTH];
        cout << "Enter the device type or substring to search for: ";
        cin.ignore();
        cin.getline(searchString, MAX_STRING_LENGTH);

        for (int i = 0; i < inputtedRequests; i++)
        {
```

```
            if (strstr(reqArray[i].deviceType, searchString) != nullptr)
            {
                foundRequests[foundCount++] = reqArray[i];
            }
        }

        if (foundCount > 0)
        {
            DisplayDevices(foundRequests, foundCount);
        }
        else
        {
            cout << "No devices found matching the search criteria." << endl;
        }
    }
    else if (searchType == 1)
    {
        int searchStatus;
        cout << "Enter the request status to search for (0 for PENDING, 1 for REJECTED,
2 for COMPLETED): ";
        cin.ignore();
        cin >> searchStatus;

        for (int i = 0; i < inputtedRequests; i++)
        {
            if (reqArray[i].status == searchStatus)
            {
                foundRequests[foundCount++] = reqArray[i];
            }
        }
        if (foundCount > 0)
        {
            DisplayDevices(foundRequests, foundCount);
        }
        else
        {
            cout << "No devices found matching the search criteria." << endl;
        }
    }
}
```

5.3.1. Входни данни на функцията

Тук се представят входните данни на функцията: какви са, защо са избрани точно те, за какво ще се използват.

5.3.2. Изходни данни на функцията или данни, които се извеждат

Функцията извежда сортираните и филтрирани данни в същата таблица като условие С.

6. Реализация на условие E

6.1. Анализ на алгоритъма, който трябва да се реализира

Използва се Bubble Sort алгоритъл който сравнява първо по година после по месец после по ден.

6.2. Функция с която е реализиран алгоритъма

```
void SortByDate(Request reqArray[], int const inputtedRequests)
{
    auto parseDate = [](const char* date, int& day, int& month, int& year) {
        sscanf_s(date, "%d-%d-%d", &day, &month, &year);
        };

    for (int i = 0; i < inputtedRequests - 1; ++i) {
        for (int j = 0; j < inputtedRequests - i - 1; ++j) {
            int day1, month1, year1;
            int day2, month2, year2;
            parseDate(reqArray[j].date, day1, month1, year1);
            parseDate(reqArray[j + 1].date, day2, month2, year2);

            if (year1 > year2 || (year1 == year2 && month1 > month2) || (year1 == year2 &&
month1 == month2 && day1 > day2)) {
                Request temp = reqArray[j];
                reqArray[j] = reqArray[j + 1];
                reqArray[j + 1] = temp;
            }
        }
    }
    DisplayDevices(reqArray, inputtedRequests);
}
```

6.2.1. Изходни данни на функцията или данни, които се извеждат

Създава се нов масив в който се сортират всички заявки и той се подава на DisplayRequests(), който извежда с идентичен формат на условие C.

7. Реализация на условие F

7.1. Анализ на алгоритъма, който трябва да се реализира

След като потребителя избере дали ще чете или записва данни влизаме в съответнатия клон на функцията. При избор за записване, създаваме и отваряме файлова променлива, след което записваме всички заявки в него. Когато четем от файл обаче, първо смятаме броя заявки, като разделим размера на файла на размера на една заявка, след което четем такъв брой заявки от файла и ги въвеждаме в временен масив и ги изпращаме на DisplayRequests().

7.2. Функция с която е реализиран алгоритъма

```
void ImportAndExportBinaryFile(Request reqArray[], int& inputtedRequests)
{
    cin.ignore();
    cout << "Do you want to import data from file (0) or export data to file (1)?" << endl;
    bool fine = false;
    int command;
    do
    {
        cin >> command;
        if (command < 0 || command > 1)
        {
            cout << "Wrong input! Try Again: (0 for import, 1 for export): ";
        }
        else fine = true;
    } while (fine == false);
    if (command == 1)
    {
        ofstream output_file("requests.dat", ios::binary);
        if (output_file.is_open())
        {
            for (int i = 0; i < inputtedRequests; i++)
            {
                output_file.write((char*)&reqArray[i], sizeof(reqArray[i]));
            }
            output_file.close();
            cout << "Data successfully exported to file." << endl;
        }
        else
        {
            cerr << "Error opening file for writing." << endl;
        }
    }
    else
    {
        ifstream input_file("requests.dat", ios::binary | ios::ate);
        if (!input_file.is_open()) {
            cerr << "Error opening file for reading." << endl;
            return;
        }
        streamsize file_size = input_file.tellg();
        input_file.seekg(0, ios::beg);

        int requestCount = file_size / sizeof(Request);

        if (requestCount == 0) {
            cerr << "No data in file or file size mismatch." << endl;
            input_file.close();
            return;
```

```
        }

        input_file.read(reinterpret_cast<char*>(reqArray), file_size);
        if (!input_file) {
            cerr << "Error reading data from file." << endl;
            input_file.close();
            return;
        }

        input_file.close();
        cout << "Data successfully imported from file." << endl;
        inputtedRequests += requestCount;
    }
}
```

7.2.1. Входни данни на функцията

Входни данни е единствено изборът дали да се пише във файл или да се чете.

7.2.2. Изходни данни на функцията или данни, които се извеждат

Извежда се таблица с всички прочетени заявки. Извежда се съобщение за успешно записване. Извеждат се съобщения при грешка.

8. Реализация на условие G - допълнение първо

8.1. Анализ на алгоритъма, който трябва да се реализира

Функцията AdditionalRequestInformation има за цел да предостави допълнителна информация за заявките, като позволява на потребителя да избира между две опции: показване на завършени заявки за определен техник или показване на всички заявки с определен проблем за определено устройство. Ето описание на алгоритъма:

Извеждане на меню и въвеждане на команда:

Функцията започва с извеждане на меню, което предлага две опции на потребителя.

Използва се cin.ignore() за изчистване на входния буфер.

Потребителят въвежда команда (0 или 1), която се проверява в цикъл do-while, докато не бъде въведена валидна команда.

Обработка на команда 1:

Ако потребителят избере команда 1, се извършва търсене на заявки по устройство и проблем.

Потребителят въвежда подниз за търсене на тип устройство и описание на проблема.

Създава се масив filteredRequests за съхранение на филтрираните заявки и брояч filteredCount.

Всички заявки в reqArray се преглеждат и тези, които съдържат въведения подниз за устройство и проблем, се добавят към filteredRequests.

Филтрираните заявки се сортират по тип устройство чрез функцията SortByDeviceType и се извеждат чрез функцията DisplayDevices.

Обработка на команда 0:

Ако потребителят избере команда 0, се извършва търсене на завършени заявки за определен техник.

Създава се масив completedRequests за съхранение на завършените заявки и брояч completedCount.

Всички заявки в reqArray се преглеждат и тези със статус COMPLETED се добавят към completedRequests.

Извеждат се имената на техниците за завършените заявки.

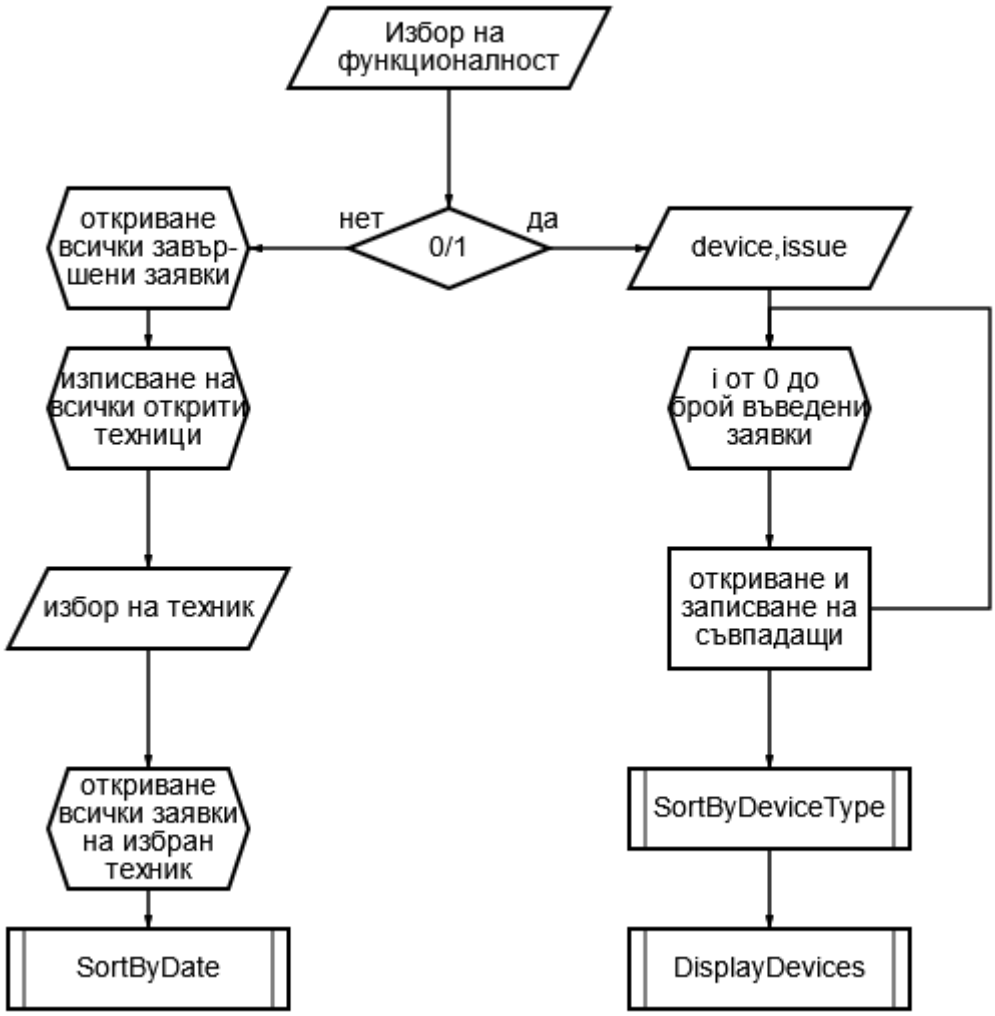
Потребителят въвежда подниз за търсене на име на техник.

Създава се масив filteredRequests за съхранение на филтрираните заявки и брояч filteredCount.

Всички завършени заявки се преглеждат и тези, които съдържат въведения подниз за име на техник, се добавят към filteredRequests.

Филтрираните заявки се сортират по дата чрез функцията SortByDate.

8.2. Блок схема на алгоритъма



8.3. Функция с която е реализиран алгоритъма

```
void AdditionalRequestInformation(Request reqArray[], int& inputtedRequests)
{
    cout << "To displaying completed requests for certain technician (0). \n To display all requests with a certain issue for a certain device (1)\n: ";
    cin.ignore();
    int command;
    do
    {
        cin >> command;
        if (command > 1 || command < 0 || cin.fail())
        {
            cout << "Wrong Command. Please try again. To displaying completed requests for certain technician (0). \n To display all requests with a certain issue for a certain device (1)\n: ";
        }
        else break;
    } while (true);
```

```
    if (command == 1)
    {
        char searchDevice[MAX_STRING_LENGTH];
        cout << "Enter a device type substring to search for: ";
        cin.ignore();
        cin.getline(searchDevice, MAX_STRING_LENGTH);
        char searchIssue[MAX_STRING_LENGTH];
        cout << "Enter an issue description substring to search for: ";
        cin.ignore();
        cin.getline(searchIssue, MAX_STRING_LENGTH);
        Request filteredRequests[MAX_LENGTH];
        int filteredCount = 0;
        for (int i = 0; i < inputtedRequests; ++i)
        {
            if (strstr(reqArray[i].deviceType, searchDevice) != nullptr &&
strstr(reqArray[i].issueDescription, searchIssue) != nullptr)
            {
                filteredRequests[filteredCount++] = reqArray[i];
            }
        }
        SortByDeviceType(filteredRequests, filteredCount);
        DisplayDevices(filteredRequests, filteredCount);
    }
    else if (command == 0)
        {
            Request completedRequests[MAX_LENGTH];
            int completedCount = 0;

            for (int i = 0; i < inputtedRequests; ++i)
            {
                if (reqArray[i].status == COMPLETED)
                {
                    completedRequests[completedCount++] = reqArray[i];
                }
            }

            cout << "Technician Names for Completed Requests:" << endl;
            for (int i = 0; i < completedCount; ++i)
            {
                cout << completedRequests[i].technicianName << endl;
            }

            char searchTechnician[MAX_STRING_LENGTH];
            cout << "Enter a technician name substring to search for: ";
            cin.ignore();
            cin.getline(searchTechnician, MAX_STRING_LENGTH);

            Request filteredRequests[MAX_LENGTH];
            int filteredCount = 0;
```

```
        for (int i = 0; i < completedCount; ++i)
        {
            if (strstr(completedRequests[i].technicianName, searchTechnician) != nullptr)
            {
                filteredRequests[filteredCount++] = completedRequests[i];
            }
        }
        SortByDate(filteredRequests, filteredCount);
    }
}
```

8.3.1. Входни данни на функцията

Потребителя избира каква допълнителна функционалност иска от конзолата, след което ако избере да търси по у-во и проблем бива запитан да избере съответни такива, а ако избере да провери всички заявки на техник трябва да избере името на техника от изведените такива.

8.3.2. Изходни данни на функцията или данни, които се извеждат

```
To displaying completed requests for certain technician (0).
 To display all requests with a certain issue for a certain device (1)
: 0
Technician Names for Completed Requests:
Robert Ratakowski
Jordan Philipow
Jordan Philipow
Enter a technician name substring to search for:
```

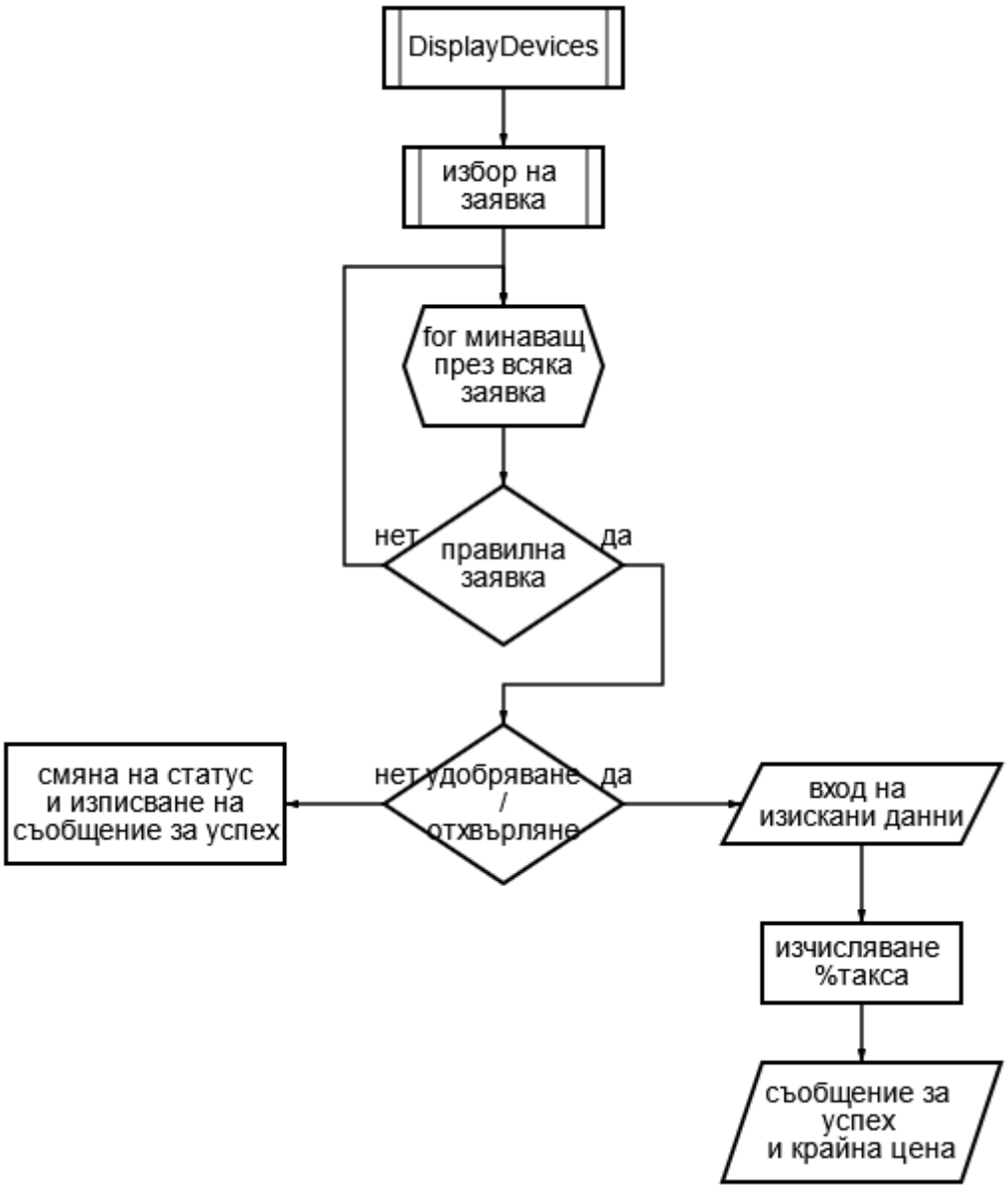
```
: 1
Enter a device type substring to search for: p
Enter an issue description substring to search for: i
Request  Date       Client Name       Device Type    Serial No     Issue Description       Technician Name    Repair Details         CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------
3        12-09-2019 Mihaela Filipova  Laptop         ijhguig       Software crash          Jordan Philipow    einstalled             180 COMPLETED FAST
Please input command from 1 to 9.
```

9. Реализация на условие H,I - допълнение второ

9.1. Анализ на алгоритъма, който трябва да се реализира

Комбинирайки двете подусловия в едно алгоритъма приема като първа променлива дали потребителя ще откаже или приключи заявка. Ако реши да приключи заявка трябва да попълни данните на техника, поправката и цената, която съответно бива сметната и изписана като крайна цена за потребителя. Статусът се променя от потръвдена на завършена. Ако поръчка бъде отказана статусът и се сменя и не се изисква нищо допълнително. Заявка може да бъде завършена и да и се променя статусът само и единствено ако тя е със статус приета.

9.2. Блок схема на алгоритъма



9.3. Функция с която е реализиран алгоритъма

```
void CompleteRequest(Request reqArray[], int const inputtedRequests)
{
    int requestNumber;
    DisplayDevices(reqArray, inputtedRequests);
    cout << "Enter the request number you want to complete: ";
    cin >> requestNumber;
    for (int i = 0; i < inputtedRequests; i++)
    {
        if (reqArray[i].requestNumber == requestNumber)
        {
            if (reqArray[i].status == 0)
            {
                cout << "To COMPLETE request (0). To REJECT request (1); ";
```

```
            int input;
            do
            {
                cin >> input;
                if (cin.fail() || input > 2 || input < 0)
                {
                    cout << "Wrong input please try again: To COMPLETE request (0). To
REJECT request (1): ";
                }
                else break;
            } while (true);

            if (input == 1)
            {
                reqArray[i].status = static_cast<RequestStatus>(1);
                cout << "Request successfully REJECTED." << endl;
                return;
            }
            else
            {
                cout << "Enter Technician Name: ";
                cin.getline(reqArray[i].technicianName, MAX_STRING_LENGTH);

                cout << "Enter Repair Details: ";
                cin.getline(reqArray[i].repairDetails, MAX_STRING_LENGTH);

                do
                {
                    cout << "Enter Repair Cost before type calculation: ";
                    cin >> reqArray[i].repairCost;
                    if (cin.fail())
                    {
                        cout << "Try again please." << endl;
                    }
                    else break;
                } while (true);

                if (reqArray[i].type == 0)
                {
                    cout << "Final price is: " << reqArray[i].repairCost << "because request
type is NORMAL" << endl;
                }
                else if (reqArray[i].type == 1)
                {
                    reqArray[i].repairCost += reqArray[i].repairCost / 5;
                    cout << "Final price is: " << reqArray[i].repairCost << "because request
type is FAST" << endl;
                }
                else
```

```
                {
                    reqArray[i].repairCost += reqArray[i].repairCost / 2;
                    cout << "Final price is: " << reqArray[i].repairCost << "because request
type is EXPRESS" << endl;
                }

                reqArray[i].status = static_cast<RequestStatus>(2);
                cout << "Request successfully completed." << endl;
                return;
            }
        }
        else cout << "Request has either been COMPLETED or REJECTED" << endl;
        return;
    }
  }
  cout << "Request number " << requestNumber << " not found." << endl;
}
```

9.3.1. Входни данни на функцията

От потребителя се изисква да избере заявка, след това дали да я завърши или откаже. Ако я завърши ще трябва да въведе изисканите данни(Име на техник, Извършена процедура, Цена преди такса).

9.3.2. Изходни данни на функцията или данни, които се извеждат

Първо биват изведени всички приети заявки, след което се извеждат приканванията за въвеждане и накрая съобщение при успех. При липса на търсената заявка се изписва съобщение.

```
Request  Date       Client Name       Device Type    Serial No     Issue Description       Technician Name    Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob        Phone          lej928f       Battery issue           Robert Ratakowski  ew battery installed    150 COMPLETED NORMAL
2        21-12-2023  Eduard Ivanov     Tablet         asjdo9q       Overheating             PENDING            PENDING                 0   REJECTED  EXPRESS
3        12-09-2019  Mihaela Filipova  Laptop         ijhguig       Software crash          Jordan Philipow    einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski  Phone          khugug        No power                Jordan Philipow    ew battery installed    240 COMPLETED FAST
Enter the request number you want to complete: 2
Request has either been COMPLETED or REJECTED
```

10. Реализация на допълнение трето

10.1. Анализ на алгоритъма, който трябва да се реализира

Вика се аналогичен алгоритъм на този на условие F като в началото на файла се вика с параметър който вкарва функцията в статус на първоначално зареждане на данни и преди да се прекрати след напускане с въвеждане на число 9 в конзолата се извиква отново функцията за автоматична синхронизация на данните, с параметър сочещ към затваряне на програмата, тоест данните от основния масив трябва да бъдат записани обратно във файла. Тази функция не чака вход, записва в отделен файл и единственият изход е дали записването/четенето е успешно или не.

10.2. Блок схема на алгоритъма

10.3. Функция с която е реализиран алгоритъма

```
void ImportAndExportBinaryFileAutomatically(Request reqArray[], int&
inputtedRequests, bool end)
{
    if (end == false)
    {
        ifstream input_file("requestsPermanent.dat", ios::binary | ios::ate);
        if (!input_file.is_open()) {
            cerr << "Error opening file for reading." << endl;
            return;
        }
        streamsize file_size = input_file.tellg();
        input_file.seekg(0, ios::beg);

        int requestCount = file_size / sizeof(Request);

        if (requestCount == 0) {
            cerr << "No data in file or file size mismatch." << endl;
            input_file.close();
            return;
        }

        input_file.read(reinterpret_cast<char*>(reqArray), file_size);
        if (!input_file) {
            cerr << "Error reading data from file." << endl;
            input_file.close();
            return;
        }

        input_file.close();
        cout << "Data successfully imported from file." << endl;
        inputtedRequests += requestCount;
    }
    else
    {
```

```
    ofstream output_file("requestsPermanent.dat", ios::binary);
    if (output_file.is_open())
    {
        for (int i = 0; i < inputtedRequests; i++)
        {
            output_file.write((char*)&reqArray[i], sizeof(reqArray[i]));
        }
        output_file.close();
        cout << "Data successfully exported to file." << endl;
    }
    else
    {
        cerr << "Error opening file for writing." << endl;
    }
  }
}
```

10.3.1. Изходни данни на функцията или данни, които се извеждат

```
Data successfully imported from file.
Please input command from 1 to 9.
```

II. Упътване за употреба

1. Стартиране на приложението

2. Меню с команди

1. Въведете нова заявка
2. Показване на всички устройства
3. Търсене на заявка
4. Сортиране по дата
5. Импортиране/Експортиране на заявки
6. Завършване на заявка
7. Допълнителна информация за заявка
8. Изчистване на екрана и показване на менюто
9. Изход

3. Избор на команда

- Въведете номер на команда (1-9) и натиснете Enter.
- Ако въведете невалидна команда, ще получите съобщение "Wrong command!".

4. Описание на командите и очаквани входни данни

-

1. Въведете нова заявка: Позволява ви да въведете нова заявка в системата.

- **Очаквани входни данни**: информация за нова заявка (например, дата, клиентско име, тип устройство, сериен номер, описание на проблема). ID на заявката се попълва автоматично от програмата.
- **Пример**:

```
Въведете дата (DD-MM-YYYY): 01-10-2023

Въведете име на клиента: Иван Иванов

Изберете тип устройство (1-10): 1

Въведете сериен номер: ABC123456

Изберете описание на проблема (1-10): 1
```

Въведете дата (DD-MM-YYYY): 01-10-2023

Въведете име на клиента: Иван Иванов

Изберете тип устройство (1-10): 1

Въведете сериен номер: ABC123456

Изберете описание на проблема (1-10): 1

2. Показване на всички устройства: Показва списък с всички въведени устройства.

- **Очаквани входни данни**: Няма входни данни, просто показва всички устройства.

3. Търсене на заявка: Позволява ви да търсите заявка по определени критерии.

- **Очаквани входни данни**: критерии за търсене (например, тип устройство или статус на заявка).

- **Пример**:
  Търсене по тип устройство (0) или статус на заявка (1): 0

  Въведете тип устройство или подстринг за търсене: Laptop

4. Сортиране по дата: Сортира заявките по дата.

- **Очаквани входни данни**: Няма входни данни, просто сортира заявките по дата.

5. Импортиране/Експортиране на заявки: Импортира или експортира заявки от/в бинарен файл.

- **Очаквани входни данни**: избор между импортиране или експортиране на заявки.
- **Пример**:

```
Въведете 0 за импортиране или 1 за експортиране: 0
```

Въведете 0 за импортиране или 1 за експортиране: 0

6. Завършване на заявка: Маркира заявка като завършена.

- **Очаквани входни данни**: ID на заявка, която да бъде завършена.
- **Пример**:

```
Въведете ID на заявка за завършване: 101
```

Въведете ID на заявка за завършване: 101

7. Допълнителна информация за заявка: Показва допълнителна информация за избрана заявка.

- **Очаквани входни данни**: избор между показване на завършени заявки за определен техник или всички заявки с определен проблем за определено устройство.
- **Пример:**

Въведете 0 за показване на завършени заявки за определен техник или 1 за всички заявки с определен проблем за определено устройство: 0

8. Изчистване на екрана и показване на менюто: Изчиства екрана и показва менюто отново.

Очаквани входни данни: Няма входни данни, просто изчиства екрана и показва менюто отново.

5. Изход

- За да излезете от приложението, въведете команда 9 и натиснете Enter.
- Приложението автоматично ще запази текущото състояние на заявките в бинарен файл.

III. Примерно действие на програмата

1. **Стартиране на приложението**

```
./DSR
```

2. **Меню с команди**

```
1. Въведете нова заявка
2. Показване на всички устройства
3. Търсене на заявка
4. Сортиране по дата
5. Импортиране/Експортиране на заявки
6. Завършване на заявка
7. Допълнителна информация за заявка
8. Изчистване на екрана и показване на менюто
9. Изход
```

3. **Избор на команда**

```
Въведете команда: 1
```

4. **Въвеждане на нова заявка**

```
Въведете дата (DD-MM-YYYY): 01-10-2023
Въведете име на клиента: Иван Иванов
Изберете тип устройство (1-10): 1
Въведете сериен номер: ABC123456
Изберете описание на проблема (1-10): 1
```

5. **Показване на всички устройства**

```
Въведете команда: 2
```

6. **Търсене на заявка**

```
Въведете команда: 3
Търсене по тип устройство (0) или статус на заявка (1): 0
Въведете тип устройство или подстринг за търсене: Laptop
```

7. **Сортиране по дата**

```
Въведете команда: 4
```

8. **Импортиране/Експортиране на заявки**

```
Въведете команда: 5
Въведете 0 за импортиране или 1 за експортиране: 0
```

9. **Завършване на заявка**

```
Въведете команда: 6
Въведете ID на заявка за завършване: 101
```

10. **Допълнителна информация за заявка**

```
Въведете команда: 7
Въведете 0 за показване на завършени заявки за определен техник или 1 за всички заявки с определен пробл
```

11. **Изчистване на екрана и показване на менюто**

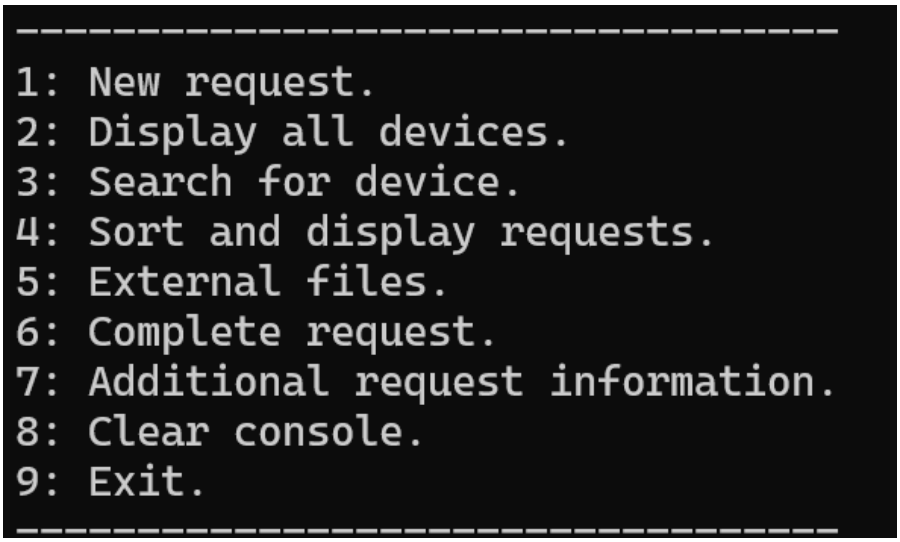
```
Въведете команда: 8
```

12. **Изход от приложението**

```
Въведете команда: 9
```

Следвайте тези стъпки, за да използвате ефективно конзолното приложение за управление на заявки.

1. Условие А


1: New request.
2: Display all devices.
3: Search for device.
4: Sort and display requests.
5: External files.
6: Complete request.
7: Additional request information.
8: Clear console.
9: Exit.

2. Условие В

```
Please input command from 1 to 9.
1
Please enter the number of requests you want to input:1
Request Number is: 5
Enter Date (DD-MM-YYYY): 12-12-2024
Enter Client Name: Iordan Petkov
1: Laptop
2: Phone
3: Tablet
4: Desktop
5: Smartwatch
6: Printer
7: Router
8: Camera
9: Monitor
10: Keyboard
Select Device Type: 1
Enter Serial Number: kjbhugh
1: Screen not working
2: Battery issue
3: Overheating
4: Software crash
5: Hardware failure
6: Network issue
7: Slow performance
8: No power
9: Sound issue
10: Keyboard malfunction
Select Issue Description: 1
Technician Name is pending.
Repair Details are pending.
Repair Cost is pending.
Enter Status (0 for NORMAL, 1 for FAST, 2 for EXPRESS): 1

Request data successfully entered!

---------------------------------
```

3. Условие C

```
3
Search for request by device type(0) or request status(1).
0
Enter the device type or substring to search for: Ph
Request  Date        Client Name        Device Type    Serial No     Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob         Phone          lej928f       Battery issue           Robert Ratakowski   ew battery installed    150 COMPLETED NORMAL
4        3-07-2024   Ivan Kartagenski   Phone          khugug        No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
Please input command from 1 to 9.
3
Search for request by device type(0) or request status(1).
1
Enter the request status to search for (0 for PENDING, 1 for REJECTED, 2 for COMPLETED): 2
Request  Date        Client Name        Device Type    Serial No     Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob         Phone          lej928f       Battery issue           Robert Ratakowski   ew battery installed    150 COMPLETED NORMAL
3        12-09-2019  Mihaela Filipova   Laptop         ijhguig       Software crash          Jordan Philipow     einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski   Phone          khugug        No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
Please input command from 1 to 9.
```

4. Условие D

```
4
Request  Date        Client Name        Device Type    Serial No     Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
3        12-09-2019  Mihaela Filipova   Laptop         ijhguig       Software crash          Jordan Philipow     einstalled              180 COMPLETED FAST
2        21-12-2023  Eduard Ivanov      Tablet         asjdo9q       Overheating             PENDING             PENDING                 0   REJECTED  EXPRESS
1        12-02-2024  Robert Bob         Phone          lej928f       Battery issue           Robert Ratakowski   ew battery installed    150 COMPLETED NORMAL
4        3-07-2024   Ivan Kartagenski   Phone          khugug        No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
5        12-12-2024  Iordan Petkov      Laptop         kjbhugh       Screen not working      PENDING             PENDING                 0   PENDING   FAST
Please input command from 1 to 9.
```

5. Условие E

```
5
Do you want to import data from file (0) or export data to file (1)?
0
Data successfully imported from file.
Please input command from 1 to 9.
5
Do you want to import data from file (0) or export data to file (1)?
1
Data successfully exported to file.
Please input command from 1 to 9.
```

6. Условие F

```
7
To displaying completed requests for certain technician (0).
 To display all requests with a certain issue for a certain device (1)
: 0
Technician Names for Completed Requests:
Robert Ratakowski
Jordan Philipow
Jordan Philipow
Enter a technician name substring to search for: Jo
Request  Date        Client Name        Device Type    Serial No     Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
3        12-09-2019  Mihaela Filipova   Laptop         ijhguig       Software crash          Jordan Philipow     einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski   Phone          khugug        No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
Please input command from 1 to 9.
7
To displaying completed requests for certain technician (0).
 To display all requests with a certain issue for a certain device (1)
: 1
Enter a device type substring to search for: Ph
Enter an issue description substring to search for: i
Request  Date        Client Name        Device Type    Serial No     Issue Description       Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob         Phone          lej928f       Battery issue           Robert Ratakowski   ew battery installed    150 COMPLETED NORMAL
4        3-07-2024   Ivan Kartagenski   Phone          khugug        No power                Jordan Philipow     ew battery installed    240 COMPLETED FAST
Please input command from 1 to 9.
```

7. Допълнение първо и второ

```
6
Request  Date        Client Name         Device Type    Serial No      Issue Description        Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob          Phone          lej928f        Battery issue            Robert Ratakowski   ew battery installed    150 COMPLETED NORMA
2        21-12-2023  Eduard Ivanov       Tablet         asjdo9q        Overheating              PENDING             PENDING                 0   REJECTED  EXPRE
3        12-09-2019  Mihaela Filipova    Laptop         ijhguig        Software crash           Jordan Philipow     einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski    Phone          khugug         No power                 Jordan Philipow     ew battery installed    240 COMPLETED FAST
5        12-12-2024  Ivan Kostov         Phone          ;klkajdiuuwqh  Screen not working       PENDING             PENDING                 0   PENDING   FAST
6        2-12-2024   Yordan Hristov      Laptop         ;lmlkjjnh      Keyboard malfunction     PENDING             PENDING                 0   PENDING   NORMA
Enter the request number you want to complete: 5
To COMPLETE request (0). To REJECT request (1); 1
Request successfully REJECTED.
Please input command from 1 to 9.
6
Request  Date        Client Name         Device Type    Serial No      Issue Description        Technician Name     Repair Details          CostStatus    Type
-------------------------------------------------------------------------------------------------------------------------------------------------------------
1        12-02-2024  Robert Bob          Phone          lej928f        Battery issue            Robert Ratakowski   ew battery installed    150 COMPLETED NORMA
2        21-12-2023  Eduard Ivanov       Tablet         asjdo9q        Overheating              PENDING             PENDING                 0   REJECTED  EXPRE
3        12-09-2019  Mihaela Filipova    Laptop         ijhguig        Software crash           Jordan Philipow     einstalled              180 COMPLETED FAST
4        3-07-2024   Ivan Kartagenski    Phone          khugug         No power                 Jordan Philipow     ew battery installed    240 COMPLETED FAST
5        12-12-2024  Ivan Kostov         Phone          ;klkajdiuuwqh  Screen not working       PENDING             PENDING                 0   REJECTED  FAST
6        2-12-2024   Yordan Hristov      Laptop         ;lmlkjjnh      Keyboard malfunction     PENDING             PENDING                 0   PENDING   NORMA
Enter the request number you want to complete: 6
To COMPLETE request (0). To REJECT request (1); 0
Enter Technician Name: Iordan Filipow
Enter Repair Details: New keyboard installed
Enter Repair Cost before type calculation: 250
Final price is: 250because request type is NORMAL
Request successfully completed.
```

8. Допълнение трето

```
int main()
{
    PrintMenu();
    const int MAX_REQUESTS = 100;
    Request reqArray[MAX_REQUESTS];
    int inputtedRequests = 0;
    ImportAndExportBinaryFileAutomatically(reqArray, inputtedRequests, false);
    int command;
    do
    {
        command = InputCommand(command);

        if (command < 1 || command > 9 || cin.fail())
        {
            cout << "Wrong command!" << endl;
        }
        else
        {
            switch (command)
            {
            case 1: InputRequest(reqArray, inputtedRequests); break;
            case 2: DisplayDevices(reqArray, inputtedRequests); break;
            case 3: SearchForRequest(reqArray, inputtedRequests); break;
            case 4: SortByDate(reqArray, inputtedRequests); break;
            case 5: ImportAndExportBinaryFile(reqArray, inputtedRequests); break;
            case 6: CompleteRequest(reqArray, inputtedRequests); break;
            case 7: AdditionalRequestInformation(reqArray, inputtedRequests); break;
            case 8: system("cls"); PrintMenu(); break;
            }
        }
    } while (command != 9);
    ImportAndExportBinaryFileAutomatically(reqArray, inputtedRequests, true);
}
```

```
---------------------------------
1: New request.
2: Display all devices.
3: Search for device.
4: Sort and display requests.
5: External files.
6: Complete request.
7: Additional request information.
8: Clear console.
9: Exit.
---------------------------------
Data successfully imported from file.
Please input command from 1 to 9.
```