

РУСЕНСКИ УНИВЕРСИТЕТ „АНГЕЛ КЪНЧЕВ“

Катедра: Компютърни системи и технологии

КУРСОВА ЗАДАЧА

по дисциплината

„Синтез и анализ на алгоритми“

Изготвил:

фак. номер:

група:

курс:

специалност:

Дата:

Приел:.....

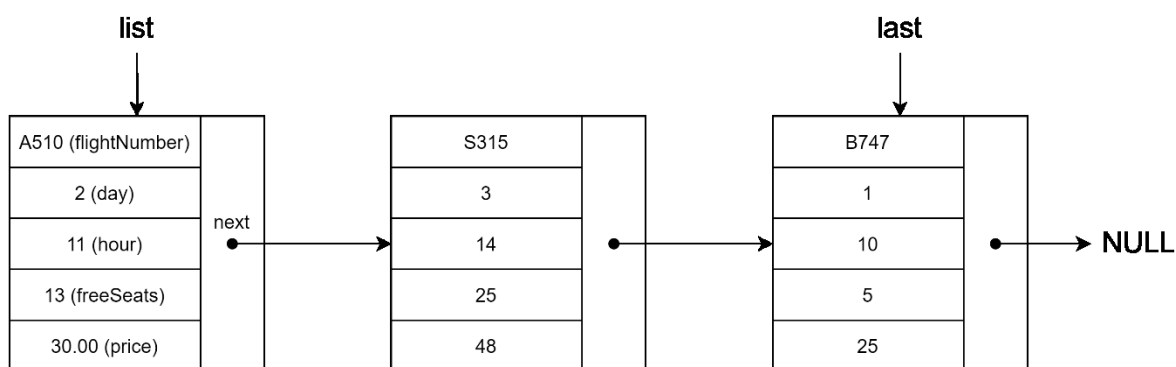
/доц. д-р Е. Големанова/

2. Задание

№ 8: Авиокомпания съхранява справочната си информация за една седмица в списък, всеки елемент на който съдържа следните данни: номер на полет, ден от седмицата, час на излитане, брой свободни места, цена на един билет. Да се напишат функции за:

- създаване на списъка;
- промяна на разписанието на зададен полет;
- извеждане на броя на свободните места и цената на билетите за следващите за деня (след определен час) полети.

3. Графично изображение на динамичния списък с конкретни примерни стойности



4. Описание на алгоритмите, съответстващи на отделните подточки от заданието:

а. Създаване на списъка

За създаване на списъка е създадена функция `createList()`, която връща указател към първия елемент на списъка. Използват се два указателя `list` и `last` съответно към първия и към последния елемент. Добавянето на елементи се извършва в `while` цикъл, където се създават елементи и потребителят въвежда техните стойности. Ако списъкът е празен, както е случаят в началото, указателите `list` и `last` се насочват към новия елемент. Иначе новият елемент се свързва след последния и указателят `last` се насочва към новия елемент.

б. Промяна на разписанието на зададен полет

За промяна на разписанието е създадена процедура `changeSchedule(point list)`, която проверява дали списъкът е празен (ако е така, извежда

съобщение и прекратява процедурата) и дали потребителят желае да промени деня и/или часа на излитане. В случай че желае, се иска от него да въведе номер на полет, чието разписание да бъде променено. Извиква се функцията `findByNumber(point &p, string flightNum)`, създадена да проверява дали такъв полет съществува в списъка, връща резултат от проверката и насочва указателя за текущия елемент към намерения, като обхожда последователно елементите чрез преместване на указателя за текущия елемент към следващ, докато списъкът не свърши. Ако полетът не е открит, се извежда съобщение. Ако е открит в списъка, се извикват помощните процедури `changeDay(point &p)` и/или `changeHour(point &p)`, чрез които потребителят въвежда новата информация за разписанието. Ако потребителят е направил промени в разписанието на полет, се извиква процедурата `printInfo(point p)`, която извежда в конзолата информацията за един полет, в случая – за този с променено разписание.

с. Извеждане на броя на свободните места и цената на билетите за следващите за деня (след определен час) полети

За извеждане на броя на свободните места и цената на билетите за полетите в даден ден след определен час е създадена процедура `printUpcomingPerDay(point list)`. Тя проверява дали списъкът е празен и ако е така, прекратява процедурата. Ако потребителят желае да получи информация за свободните места и цената на билетите за определен ден от седмицата след даден час, се иска от него да въведе ден и час. След това булева променлива служи да запазим информация дали има полет, за който да изведем информация. В `while` цикъл се извършва проверка дали денят на елемента от списъка съответства на посочения от потребителя и дали часът е по-късен от посочения. Ако това е така, се извиква процедурата `printInfo(point p)`, която извежда в конзолата информацията за един полет, в случая – този, за който е установено, че отговаря на изискванията, и се задава стойност `true` на булевата променлива. Обхождането става, като указателят за текущия елемент се премества към следващ, докато списъкът не свърши (т.е. докато указателят не получи стойност `NULL`). Ако няма полети, които да отговарят на зададените от потребителя критерии, се извежда съобщение.

5. Описание на използваните функции

Заглавна част на функцията: `point createList()`

Действие: функция, която служи за създаване на едносвързан списък

Параметри: няма

Използвани глобални променливи: няма

Извиквани функции: няма

Заглавна част на функцията: `void changeDay(point &p)`

Действие: процедура, чрез която се променя денят в разписанието на даден полет

Параметри: указател `p` (входно-изходен параметър) към елемент от едносвързания списък

Използвани глобални променливи: няма

Извиквани функции: няма

Заглавна част на функцията: `void changeHour(point &p)`

Действие: процедура, чрез която се променя часът в разписанието на даден полет

Параметри: указател `p` (входно-изходен параметър) към елемент от едносвързания списък

Използвани глобални променливи: няма

Извиквани функции: няма

Заглавна част на функцията: `void printInfo(point p)`

Действие: процедура, която извежда в конзолата информацията за даден полет, елемент от едносвързания списък

Параметри: указател `p` към елемент от едносвързания списък

Използвани глобални променливи: `week[p->day]` (извикване на елемент по индекс от глобалния масив `const char* week[] = { "", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };`)

Извиквани функции: няма

Заглавна част на функцията: `bool findByNumber(point &p, string flightNum)`

Действие: функция, връщаща резултат от търсене на даден полет по номер в списъка (true/false, съответно успешно или неуспешно откриване) и преместваща указателя `p` към намерения елемент

Параметри: указател `p` (входно-изходен параметър) към елемент от едносвързания списък

символен низ `flightNum`

Използвани глобални променливи: няма

Извиквани функции: няма

Заглавна част на функцията: `void changeSchedule(point list)`

Действие: процедура, която служи за промяна на разписанието на полет

Параметри: указател `list` към първия елемент на списъка

Използвани глобални променливи: няма

Извиквани функции: `findByNumber(p, flightNum)`

`changeDay(p);`

`changeHour(p);`

`printInfo(p);`

Заглавна част на функцията: `void printUpcomingPerDay(point list)`

Действие: процедура за извеждане в конзолата на полетите за даден ден от седмицата след определен час

Параметри: указател list към първия елемент на списъка

Използвани глобални променливи: week[day] (извикване на елемент по индекс от глобалния масив `const char* week[] = { "", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };)`

Извиквани функции: `printInfo(p);`

Заглавна част на функцията: `int main()`

Действие: главна функция за стартиране на приложението

Параметри: няма

Използвани глобални променливи: няма

Извиквани функции: `point list = createList();`
`changeSchedule(list);`
`printUpcomingPerDay(list);`

6. Тестови примери (снимки на екрана), доказващи коректността на програмата

Тест № 1

```
Microsoft Visual Studio Debug Console
Active code page: 1251
Искате ли да добавите нов полет? (Y/N)
N
В списъка няма въведени полети!

C:\Users\user\source\repos\Coursework_SAA\Debug\203090_Tsvetelina_Stefanova_KST_25b.exe (process 3968) exited with code 0.
Press any key to close this window . . .
```

Тест № 2

```
Microsoft Visual Studio Debug Console
Искате ли да промените деня на заминаване на полет A510? (Y/N)
Y
Въведете нов ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 2
Искате ли да промените часа на заминаване на полет A510? (Y/N)
Y
Въведете нов час на излитане (цяло число между 0 и 23): 12
Данни за полета след промяната:
-----
Номер на полет: A510
Ден: Tue
Час на излитане: 12 ч.
Брой свободни места: 25
Цена на билет: 30.00 лв.
-----
Желаете ли да проверите полетите за даден ден? (Y/N)
Y
Въведете ден от седмицата (цяло число от 1 до 7): 2
Въведете след колко часа желаете да са полетите (цяло число между 0 и 23): 10
Полети в Tue след 10ч.
-----
Номер на полет: A510
Ден: Tue
Час на излитане: 12 ч.
Брой свободни места: 25
Цена на билет: 30.00 лв.
-----

C:\Users\user\source\repos\Coursework_SAA\Debug\203090_Tsvetelina_Stefanova_KST_25b.exe (process 2772) exited with code 0.
Press any key to close this window . . .
```

Тест № 3

```
Microsoft Visual Studio Debug Console
Active code page: 1251
Искате ли да добавите нов полет? (Y/N)
Y
Номер на полет: A510
Ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 3
Час на излитане (цяло число между 0 и 23): 16
Брой свободни места: 22
Цена на билета: 32
Искате ли да добавите нов полет? (Y/N)
Y
Номер на полет: S315
Ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 5
Час на излитане (цяло число между 0 и 23): 13
Брой свободни места: 10
Цена на билета: 29
Искате ли да добавите нов полет? (Y/N)
N
Искате ли да промените разписанието на полет? (Y/N)
Y
Въведете номер на полет, чието разписание желаете да промените: S315
Искате ли да промените деня на заминаване на полет S315? (Y/N)
Y
Въведете нов ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 3
Искате ли да промените часа на заминаване на полет S315? (Y/N)
Y
Въведете нов час на излитане (цяло число между 0 и 23): 15
Данни за полета след промяната:
-----
Номер на полет: S315
Ден: Wed
Час на излитане: 15 ч.
Брой свободни места: 10
Цена на билет: 29.00 лв.
-----
Желаете ли да проверите полетите за даден ден? (Y/N)
Y
Въведете ден от седмицата (цяло число от 1 до 7): 3
Въведете след колко часа желаете да са полетите (цяло число между 0 и 23): 14
Полети в Wed след 14ч.
-----
Номер на полет: A510
Ден: Wed
Час на излитане: 16 ч.
Брой свободни места: 22
Цена на билет: 32.00 лв.
-----
```

```
Microsoft Visual Studio Debug Console
Час на излитане (цяло число между 0 и 23): 13
Брой свободни места: 10
Цена на билета: 29
Искате ли да добавите нов полет? (Y/N)
N
Искате ли да промените разписанието на полет? (Y/N)
Y
Въведете номер на полет, чието разписание желаете да промените: S315
Искате ли да промените деня на заминаване на полет S315? (Y/N)
Y
Въведете нов ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 3
Искате ли да промените часа на заминаване на полет S315? (Y/N)
Y
Въведете нов час на излитане (цяло число между 0 и 23): 15
Данни за полета след промяната:
-----
Номер на полет: S315
Ден: Wed
Час на излитане: 15 ч.
Брой свободни места: 10
Цена на билет: 29.00 лв.
-----
Желаете ли да проверите полетите за даден ден? (Y/N)
Y
Въведете ден от седмицата (цяло число от 1 до 7): 3
Въведете след колко часа желаете да са полетите (цяло число между 0 и 23): 14
Полети в Wed след 14ч.
-----
Номер на полет: A510
Ден: Wed
Час на излитане: 16 ч.
Брой свободни места: 22
Цена на билет: 32.00 лв.
-----
Номер на полет: S315
Ден: Wed
Час на излитане: 15 ч.
Брой свободни места: 10
Цена на билет: 29.00 лв.
-----
C:\Users\user\source\repos\Coursework_SAA\Debug\203090_Tsvetelina_Stefanova_KST_25b.exe (process 25552) exited with code 0.
Press any key to close this window . . .
```

Тест № 4


```
Microsoft Visual Studio Debug Console
Active code page: 1251
Искате ли да добавите нов полет? (Y/N)
Y
Номер на полет: A510
Ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): 5
Час на излитане (цяло число между 0 и 23): 15
Брой свободни места: 20
Цена на билета: 40
Искате ли да добавите нов полет? (Y/N)
N
Искате ли да промените разписанието на полет? (Y/N)
N
Желаете ли да проверите полетите за даден ден? (Y/N)
Y
Въведете ден от седмицата (цяло число от 1 до 7): 5
Въведете след колко часа желаете да са полетите (цяло число между 0 и 23): 19
Полети в Fri след 19ч.
Няма полети в Fri след 19ч.!

C:\Users\user\source\repos\Coursework_SAA\Debug\203090_Tsvetelina_Stefanova_KST_25b.exe (process 25400) exited with code 0.
Press any key to close this window . . .
```

7. Разпечатка на кода

```
#include <iostream>
#include <string>
#include <cstdlib>
#include <iomanip>
using namespace std;

enum weekDay { Mon = 1, Tue, Wed, Thu, Fri, Sat, Sun };
const char* week[] = { "", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };

//описание на структурата (едносвързан списък)
struct item { string flightNumber; weekDay day; int hour; int freeSeats; double price;
item* next; };
typedef item* point;

//функция за създаване на едносвързания списък
point createList()
{
    //инициализация на указателите към първия и последния елемент
    point list = NULL;
    point last = NULL;

    char input;
    cout << "Искате ли да добавите нов полет? (Y/N)" << endl;
    cin >> input;
    while (input == 'Y')
    {
        //създаване на нов елемент
        point N = new item;
        cout << "Номер на полет: ";
        cin >> N->flightNumber;
        cout << "Ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): ";
```

```

        int d;
        cin >> d;
        N->day = (weekDay) d;
        cout << "Час на излитане (цяло число между 0 и 23): ";
        cin >> N->hour;
        cout << "Брой свободни места: ";
        cin >> N->freeSeats;
        cout << "Цена на билета: ";
        cin >> N->price;

        //проверка за празен списък
        if (list == NULL)
        {
            //насочване на указателя за началния елемент към новия елемент
            list = N;
        }
        else
        {
            //свързване на новия елемент след последния
            last->next = N;
        }

        //пренасочване на указателя за последния елемент към новия елемент
        last = N;
        //указване, че след последния елемент няма друг (край на списъка)
        N->next = NULL;

        cout << "Искате ли да добавите нов полет? (Y/N)" << endl;
        cin >> input;
    }

    //връщане на указател към входа (началния елемент) на списъка
    return list;
}

//процедура за промяна на деня в разписанието на полет
void changeDay(point &p)
{
    int d;
    cout << "Въведете нов ден от седмицата (1 = Mon, 2 = Tue, 3 = Wed, 4 = Thu, 5 = Fri, 6 = Sat, 7 = Sun): ";
    cin >> d;
    p->day = (weekDay) d;
}

//процедура за промяна на часа в разписанието на полет
void changeHour(point &p)
{
    cout << "Въведете нов час на излитане (цяло число между 0 и 23): ";
    cin >> p->hour;
}

//процедура за извеждане в конзолата на информацията за даден полет
void printInfo(point p)
{
    cout << "-----" << endl;
    cout << "Номер на полет: " << p->flightNumber << endl;
    cout << "Ден: " << week[p->day] << endl;
    cout << "Час на излитане: " << p->hour << " ч." << endl;
    cout << "Брой свободни места: " << p->freeSeats << endl;
    cout << "Цена на билет: " << setprecision(2) << fixed << p->price << " лв." << endl;
}

```

```

        cout << "-----" << endl;
    }

    //функция, връщаща резултат от търсене на даден полет по номер в списъка (true/false,
    //съответно успешно или неуспешно откриване) и преместваща указателя p към намерения
    //елемент
    bool findByNumber(point &p, string flightNum)
    {
        bool found = false;
        //цикъл за обхождане на списъка
        while (p != NULL)
        {
            //проверка за съвпадение на номера на полета с търсения номер на полет
            if (p->flightNumber == flightNum)
            {
                found = true;
                break;
            }
            //преместване на указателя за текущия елемент към следващия елемент
            p = p->next;
        }
        //връщане на резултат - успешно или неуспешно откриване на полет по даден номер
        return found;
    }

    //процедура за промяна на разписанието на полет
    void changeSchedule(point list)
    {
        //проверка дали списъкът е празен
        if (list == NULL)
        {
            cout << "В списъка няма въведени полети!" << endl;
            return;
        }

        point p = list;
        char change;
        string flightNum;
        cout << "Искате ли да промените разписанието на полет? (Y/N)" << endl;
        cin >> change;

        //проверка дали потребителят желае да промени разписанието
        if (change == 'Y')
        {
            //въвеждане на номер на полет от страна на потребителя
            cout << "Въведете номер на полет, чието разписание желаете да промените:"
";
            cin >> flightNum;
        }
        else
        {
            //прекръпяване на процедурата, ако потребителят не желае да променя
            //разписанието
            return;
        }

        //проверка за наличие на такъв полет в списъка
        if (findByNumber(p, flightNum))
        {
            char dayInput;
            cout << "Искате ли да промените деня на заминаване на полет " <<
flightNum << "? (Y/N)" << endl;

```

```

        cin >> dayInput;
        //проверка дали потребителят иска да промени деня на заминаване
        if (dayInput == 'Y')
        {
            //извикване на процедурата за промяна на деня
            changeDay(p);
        }
        char hourInput;
        cout << "Искате ли да промените часа на заминаване на полет " <<
flightNum << "? (Y/N)" << endl;
        cin >> hourInput;
        //проверка дали потребителят иска да промени часа на заминаване
        if (hourInput == 'Y')
        {
            //извикване на процедурата за промяна на часа
            changeHour(p);
        }
        if (dayInput == 'Y' || hourInput == 'Y')
        {
            cout << "Данни за полета след промяната:" << endl;
            //извикване на процедурата за печат на информацията за даден полет
при наличие на промени
            printInfo(p);
        }
    }
    else
    {
        //извеждане на съобщение, ако в списъка не е открит полет с търсения
номер
        cout << "Полетът не съществува!" << endl;
    }
}

//процедура за извеждане в конзолата на полетите за даден ден от седмицата след
определен час
void printUpcomingPerDay(point list)
{
    //проверка дали списъкът е празен
    if (list == NULL)
    {
        //прекръпяване на процедурата
        return;
    }
    weekDay day;
    int d;
    int hour;
    char check;
    cout << "Желаете ли да проверите полетите за даден ден? (Y/N)" << endl;
    cin >> check;

    //проверка дали потребителят желае да получи информация за полетите в даден ден
от седмицата след определен час
    if (check == 'Y')
    {
        //въвеждане на ден от седмицата, за който потребителят желае информация
        cout << "Въведете ден от седмицата (цяло число от 1 до 7): ";
        cin >> d;
        day = (weekDay)d;

        //въвеждане след колко часа в дадения ден от седмицата да са полетите

```

```

        cout << "Въведете след колко часа желаете да са полетите (цяло число
между 0 и 23): ";
        cin >> hour;
    }
    else
    {
        //прекръпяване на процедурата, в случай че потребителят не иска
информация за полети в даден ден след даден час
        return;
    }

    cout << "Полети в " << week[day] << " след " << hour << "ч." << endl;
    //създаване на указател за текущия елемент и насочването му към първия елемент
от списъка
    point p = list;
    //булева променлива, в която ще се пази информация дали има полети за извеждане
в конзолата
    bool exist = false;
    //цикъл за обхождане и обработка на елементите в списъка
    while (p != NULL)
    {
        //проверка дали денят съвпада с търсения и дали полетът излита след
указания час
        if (p->day == day && p->hour > hour)
        {
            //извикване на процедурата за отпечатване на информацията за даден
полет
            printInfo(p);
            //стойността на exist се променя на true, защото имаме полет,
отговарящ на условията
            exist = true;
        }
        //пренасочване на указателя за текущия елемент към следващия
        p = p->next;
    }
    if (!exist)
    {
        //извеждане на съобщение, ако в списъка няма полети, отговарящи на
критериите
        cout << "Няма полети в " << week[day] << " след " << hour << "ч.!" <<
endl;
    }
}

//главна функция
int main()
{
    system("chcp 1251");

    //създаване на списък от полети
    point list = createList();
    //извикване на процедурата за промяна на разписанието
    changeSchedule(list);
    //извикване на процедурата за извеждане на информация за полетите в даден ден от
седмичата след определен час
    printUpcomingPerDay(list);
    return 0;
}

```