Русенски университет „Ангел Кънчев“
Катедра: Компютърни системи и технологии

Курсова задача

По дисциплината
„Синтез и анализ на алгоритми“

Изготвил: Константин Христианов Ганев

Фак. Номер: 243020

Група: 6Б

Курс: 2

Специалност: КСТ

Дата: 9.10.2025г.

Приел: ……

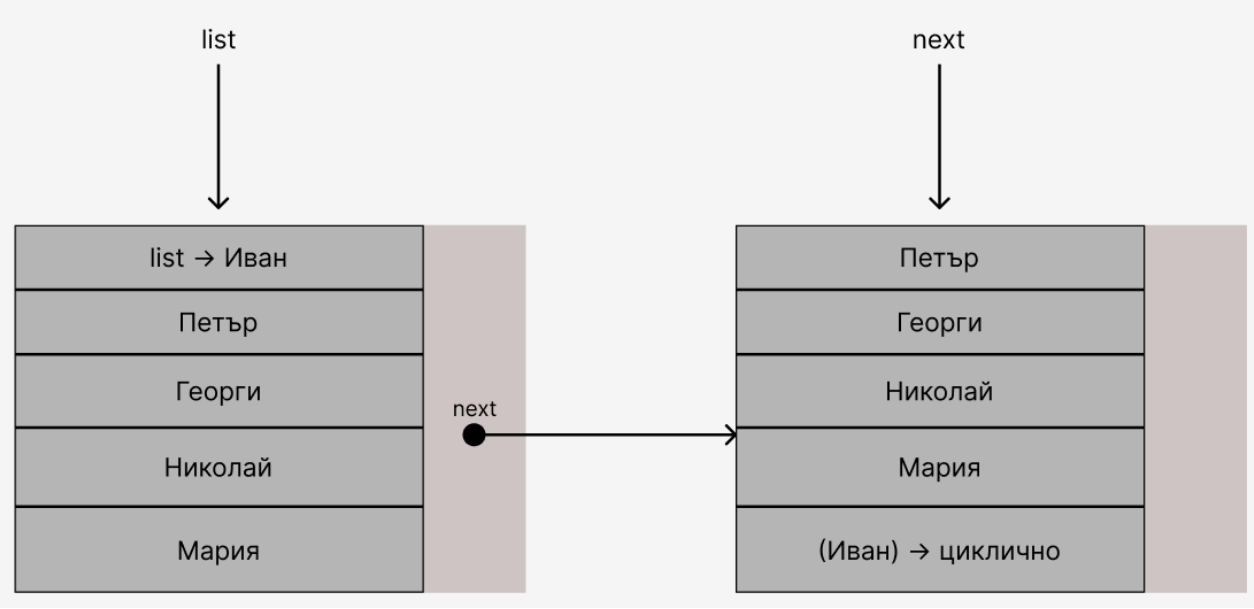
2. Задание

18. Генерал Джоузеф строил войниците в кръг, за да избере един за секретна мисия. Генералът започва да брои от произволен войник и извежда от строя всеки четвърти. Повтаряйки многократно този процес, накрая генералът избира за мисията останалия в кръга един единствен войник. Информацията за войниците (имена) се съхранява в цикличен списък.

Да се разработи програма, която:

- създава цикличен едносвързан списък, като информацията се чете от клавиатурата;
- избира един войник за мисия, а информацията за останалите се изтрива от списъка.

3. Графично изображение на динамичния списък с конкретни примерни стойности



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

а. Алгоритъм за създаване на цикличен едносвързан списък

Целта на този алгоритъм е да създаде структура от данни, в която информацията за войниците (техните имена) се съхранява последователно, като последният елемент сочи обратно към първия. Така се образува **циклична връзка**, която позволява лесно преминаване през всички войници, без да се налага връщане в началото.

Алгоритъмът се реализира по следния начин:

1. От клавиатурата се въвежда броят на войниците, които ще участват в избора.
2. За всеки войник се въвежда неговото име.
3. За всеки въведен войник се създава възел (обект) от тип Node, съдържащ:
    - име на войника (от тип string);
    - указател към следващия възел в списъка (next).
4. Първият въведен войник се свързва с втория, вторият с третия и така нататък.
5. След създаването на последния възел, неговият указател next се насочва към първия възел, с което списъкът става **цикличен**.
6. В резултат получаваме структура, в която всеки войник има свой следващ, а последният води отново към първия, образувайки непрекъснат кръг.

Тази организация на данните отразява реалното подреждане на войниците в кръг, описано в условието.

b. Алгоритъм за избор на войник за мисия и изтриване на останалите

След като цикличният списък е изграден, се реализира процесът на **избиране на войник за мисия**.
В този алгоритъм от кръг от участници последователно се премахва всеки *k*-ти, докато остане само един. В нашия случай генералът премахва всеки **четвърти** войник.

Стъпките на алгоритъма са следните:

1. Определя се начална позиция (първият войник в списъка).
2. Създават се два указателя:
    - curr – сочи към текущия войник;
    - prev – сочи към предишния войник в кръга.
3. Започва се броене на войниците. След като се преброят трима, четвъртият се извежда от строя (премахва се от списъка).
4. Изваждането от строя се реализира чрез пренасочване на указателя next на предходния възел така, че той да сочи към следващия след премахвания войник. По този начин премахнатият възел се изключва от цикъла.
5. Извеждания войник се изтрива от паметта, за да се освободи използваното място.
6. Процесът на броене продължава от следващия войник, като отново се премахва всеки четвърти.
7. Алгоритъмът приключва, когато в списъка остане само един войник. Неговото име се извежда като избрания за секретната мисия.
8. След това и последният възел се изтрива, за да се предотвратят изтичания на памет.

Резултатът от алгоритъма е името на единствения останал войник, който генералът избира за мисията, а всички останали са премахнати от списъка.

5. Описание на използваните функции

Заглавна част на функцията: Node* createList(int n);

Действие: Създава цикличен едносвързан списък от въведените войници.

Параметри:

- **int n** – брой войници, които се въвеждат.

Използвани глобални променливи:

Няма.

Извиквани функции:

Няма.

Заглавна част на функцията: string josephusSelection(Node*& last, int k);

Действие: Прилага алгоритъма и връща името на последния останал войник.

Параметри:

- **Node*& last** – указател към последния възел на списъка.
- **int k** – стъпка на елиминиране (напр. 4).

Използвани глобални променливи:

Няма.

Извиквани функции:

Няма.

Заглавна част на функцията: void deleteList(Node*& last);

Действие: Изтрива всички възли от списъка и освобождава паметта.

Параметри:

- **Node*& last** – указател към последния възел на списъка.

Използвани глобални променливи:

Няма.

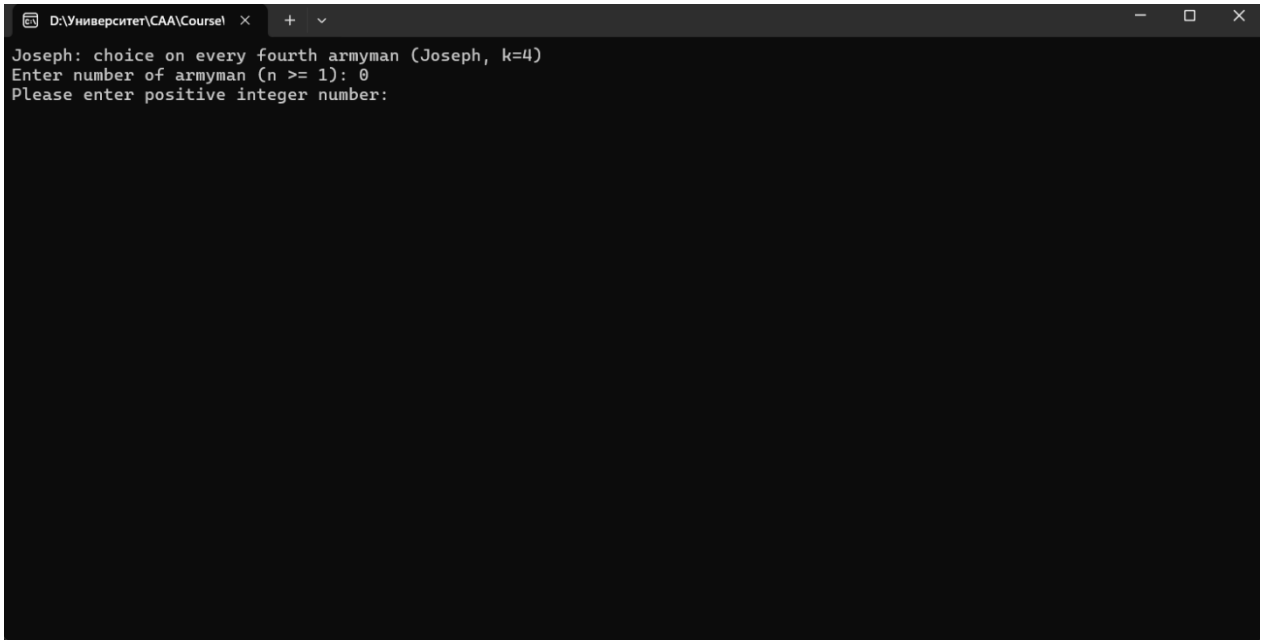
Извиквани функции:

Няма.

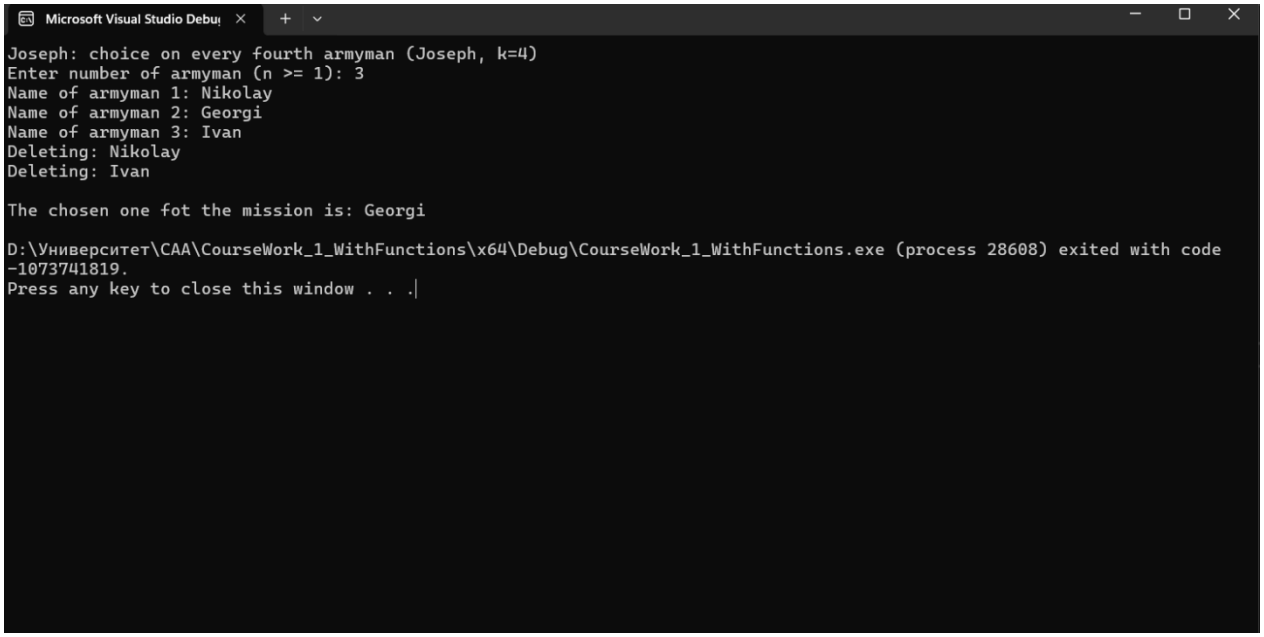
---

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

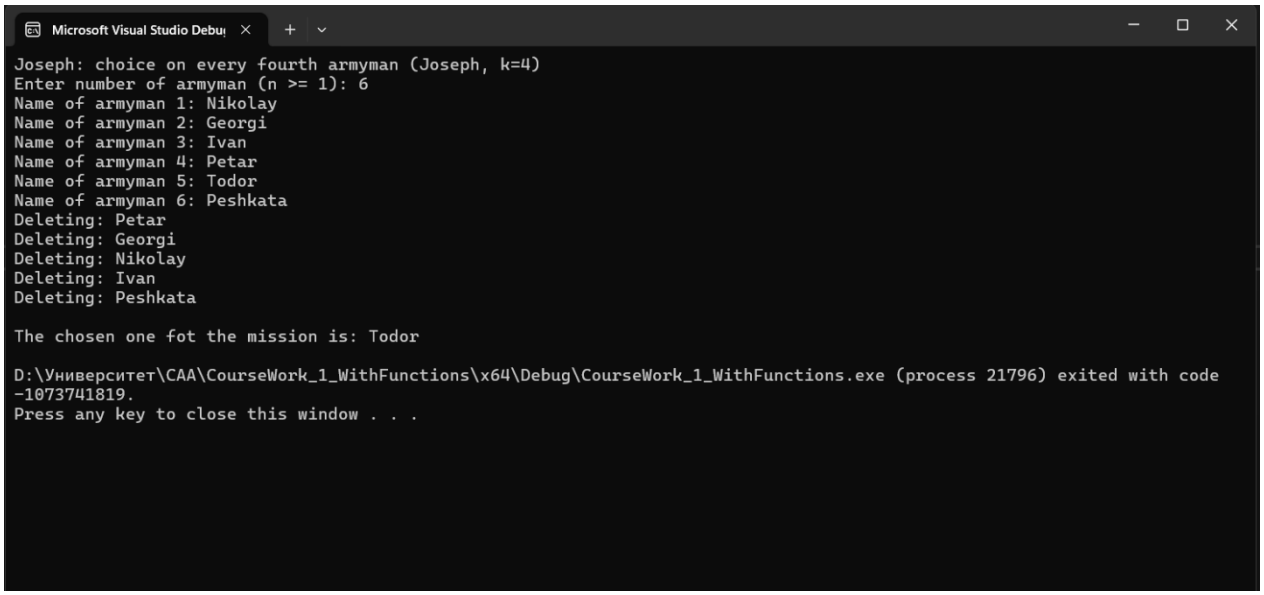
Тест №1



Тест №2


Microsoft Visual Studio Debu
Joseph: choice on every fourth armyman (Joseph, k=4)
Enter number of armyman (n >= 1): 3
Name of armyman 1: Nikolay
Name of armyman 2: Georgi
Name of armyman 3: Ivan
Deleting: Nikolay
Deleting: Ivan
The chosen one fot the mission is: Georgi
D:\Университет\CAA\CourseWork_1_WithFunctions\x64\Debug\CourseWork_1_WithFunctions.exe (process 28608) exited with code -1073741819.
Press any key to close this window . . .

Тест №3


Microsoft Visual Studio Debu
Joseph: choice on every fourth armyman (Joseph, k=4)
Enter number of armyman (n >= 1): 6
Name of armyman 1: Nikolay
Name of armyman 2: Georgi
Name of armyman 3: Ivan
Name of armyman 4: Petar
Name of armyman 5: Todor
Name of armyman 6: Peshkata
Deleting: Petar
Deleting: Georgi
Deleting: Nikolay
Deleting: Ivan
Deleting: Peshkata
The chosen one fot the mission is: Todor
D:\Университет\CAA\CourseWork_1_WithFunctions\x64\Debug\CourseWork_1_WithFunctions.exe (process 21796) exited with code -1073741819.
Press any key to close this window . . .

7. Разпечатка на кода

```
#include <iostream>
#include <string>
#include <limits>

using namespace std;

// ====================== СТРУКТУРА НА ВЪЗЕЛ ======================
struct Node {
    string name;
    Node* next;
    Node(const string& s) : name(s), next(nullptr) {}
};

// ====================== ПРОТОТИПИ НА ФУНКЦИИ ======================
Node* createList(int n);
string josephusSelection(Node*& last, int k);
void deleteList(Node*& last);

// ====================== MAIN ФУНКЦИЯ ======================
int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    cout << "Joseph: choice on every fourth armyman (Joseph, k=4)\n";

    int n;
    cout << "Enter number of armyman (n >= 1): ";
    while (!(cin >> n) || n < 1) {
        cout << "Please enter positive integer number: ";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }
    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    // 1 Създаваме цикличния списък
    Node* last = createList(n);

    // 2 Прилагаме алгоритъма на Джоузеф
    const int k = 4;
    string chosen = josephusSelection(last, k);

    // 3 Извеждаме резултата
    cout << "\nThe chosen one fot the mission is: " << chosen << "\n";

    // 4 Изтриваме остатъчния възел
    deleteList(last);

    return 0;
}

// ====================== ФУНКЦИЯ: СЪЗДАВАНЕ НА СПИСЪК ======================
Node* createList(int n) {
    Node* last = nullptr;

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

```
        cout << "Name of armyman " << (i + 1) << ": ";
        string s;
        getline(cin, s);
        if (s.empty()) { // ако натисне само Enter
            --i;
            continue;
        }

        Node* node = new Node(s);
        if (!last) {
            last = node;
            last->next = last; // първият сочи към себе си
        }
        else {
            node->next = last->next;
            last->next = node;
            last = node;
        }
    }

    return last;
}

// ===================== ФУНКЦИЯ: ИЗБОР НА ВОЙНИК ====================
string josephusSelection(Node*& last, int k) {
    if (!last)
        return "";

    Node* curr = last->next; // първият войник
    Node* prev = last;

    while (curr != prev) {
        for (int i = 1; i < k; ++i) {
            prev = curr;
            curr = curr->next;
        }

        cout << "Deleting: " << curr->name << '\n';
        prev->next = curr->next;
        delete curr;
        curr = prev->next;
    }

    string result = curr->name;
    return result;
}

// ===================== ФУНКЦИЯ: ИЗТРИВАНЕ НА СПИСЪКА ====================
void deleteList(Node*& last) {
    if (!last) return;

    Node* first = last->next;
    last->next = nullptr; // прекъсваме цикъла
    while (first) {
        Node* temp = first;
        first = first->next;
        delete temp;
```

```
    }
    last = nullptr;
}
```