

Прізвище: Пагута
Ім'я: Віталій
Група: КН-108
Варіант: 22
Дата захисту: 23.03.2022р.



Кафедра: САПР
Дисципліна: Алгоритмізація та програмування. Ч.2
Перевірів: Андрійчук М.І.

ЗВІТ

до лабораторної роботи №4
на тему "Динамічні об'єкти та способи їх використання. списки, стеки, черги"

Мета роботи: Ознайомитись із особливостями застосування динамічних об'єктів складної структури: списками, стеками та чергами; з операціями, які виконуються над елементами цих об'єктів. Набути практичних навичок програмування з використанням динамічних об'єктів складної структури.

Особливості динамічних об'єктів:

Засоби роботи з динамічною пам'ятю можуть бути реалізовані транслятором у вигляді операторів або у зовнішніх функціях. Основні властивості динамічних змінних:

- динамічні змінні створюються і видаляються програмою, що працює шляхом виконання спеціальних операторів та викликів функцій;
- кількість і розмірність динамічних змінних може змінюватись в процесі роботи програми і залежить від кількості викликів відповідних функцій і параметрів, які передаються при виклику;
- динамічна змінна не має імені, доступ до неї можливий тільки через вказівник
- при виконанні функції створення динамічної змінної в «купі», виділяється вільна пам'ять необхідного розміру і повертається вказівник на неї (адреса);
- функції видалення динамічної перемінної передається вказівник на змінну, яку видаляють. 4 Найбільш важливими властивостями динамічних змінних є їх "безіменність" і доступність по вказівнику, чим і визначається можливість варіювати число таких перемінних в програмі.

Індивідуальне завдання:

22.Сформувати однонаправлений список і видалити дві літери, які стоять перед літерою "L". Вивести на друк обидва списки.

Код програми:

```
#include <iostream>
using namespace std;

struct Text { // Створюємо односторонній список
    char Symbol;
    Text* Next;
};

Text* StartPos(Text* First, char X) { // Ініціалізує список та вставляє перший символ
    Text* T;
    T = (Text*)malloc(sizeof(Text));
    T->Symbol = X;
    T->Next = First;
    First = T;
    return First;
}

Text* RemoveFirst(Text* S) { // Забирає перший символ у списку
    Text* T;
    T = S->Next;
    free(S);
    return T;
}

Text* ToList(Text* S, char X) { // Вставляє символ у список
    Text* T;
    T = (Text*)malloc(sizeof(Text));
    T->Symbol = X;

    Text* temp = S;

    while(temp->Next)
    {
        temp = temp->Next;
    }

    T->Next = NULL;
    temp->Next = T;

    return T;
}

void Print(Text* First) { // Виводить список
    Text* T;
    T = First;
    while (T)
    {
        cout << T->Symbol << ' ';
        T = T->Next;
    }
    cout << "\n";
}

int main() {
    Text* V1, * V2, * V3, * S1, * S2, * S3;
    S1 = NULL;
    S2 = NULL;
    S3 = NULL;
```

```

V2 = S2;
V3 = S3;
char a; // Ініціалізація базових даних
cin >> a;
S1 = StartPos(S1, a); // Занесення першого символу
V1 = S1;
for (int i = 0; i < 9; i++) { // Занесення решти символів
    cin >> a;
    V1 = ToList(V1, a);
}

bool check = false;
V1 = S1;
while (V1->Next->Next->Next) {
    if (V1->Next->Next->Next->Symbol == 'l') { // Перевіряє чи закінчується на l
        if (!S2) {
            S2 = StartPos(S2, V1->Symbol);
            V1 = (V1->Next->Next->Next);
        }
        else {
            ToList(S2, V1->Symbol);
            V1 = (V1->Next->Next->Next);
        }
    }
    else {
        if (!S2) {
            S2 = StartPos(S2, V1->Symbol);
            V1 = (V1->Next);
        }
        else {
            ToList(S2, V1->Symbol);
            V1 = (V1->Next);
        }
    }
}

while (V1) {
    if (!S2) {
        S2 = StartPos(S2, V1->Symbol);
        V1 = (V1->Next);
    }
    else {
        ToList(S2, V1->Symbol);
        V1 = (V1->Next);
    }
}

cout << "List 1: "; // Виведення результатів
Print(S1);
cout << "List 2: ";
if (!S2) cout << "Empty";
else Print(S2);
}

if (check) {
    if (!S3) { // Ініціалізує S3, якщо ще не ініціалізовано та додає в S3
СИМВОЛ

```

```

        S3 = StartPos(S3, V1->Symbol);
        V3 = S3;
    }
    else { // Додає в S3 символ
        ToList(V3, V1->Symbol);
        V3 = V3->Next;
    }
}

if (V1->Symbol == 'k') { // Перевіряє чи символ рівний k
    check = true;
}

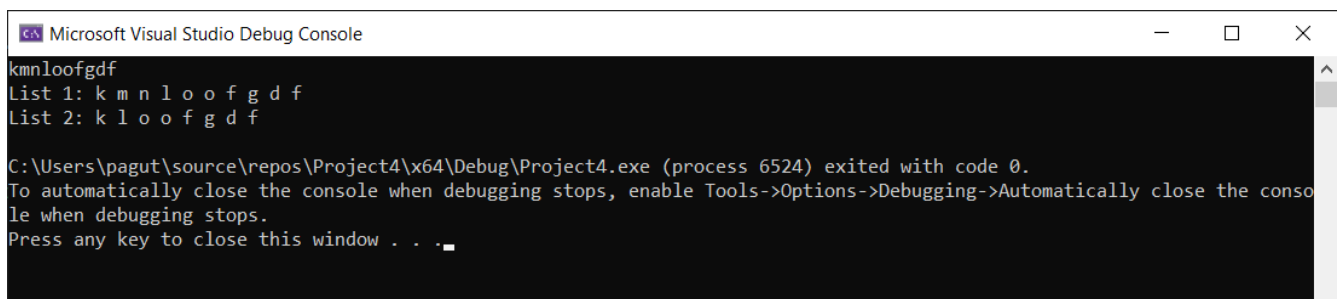
V1 = V1->Next; // Забирає перший символ
}

cout << "List 1: "; // Виведення результатів
Print(S1);

cout << "List 2: ";
if (!S2) cout << "Empty";
else Print(S2);
}

```

Результати виконання програми:



```

Microsoft Visual Studio Debug Console
kmnloofgdf
List 1: k m n l o o f g d f
List 2: k l o o f g d f

C:\Users\pagut\source\repos\Project4\x64\Debug\Project4.exe (process 6524) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .

```

Відповіді на контрольні питання:

1. Яка особливість динамічного рядка?

Довжину динамічного рядка може змінюватись в процесі роботи програми.

2. Що таке список і навіщо він використовується?

Список - це лінійна динамічна структура даних, яка складається з ланок, які зберігають корисні дані. Грубо кажучи, це є масив даних.

3. Чим відрізняється однонаправлений список від двонаправленого списку?

Однонаправлений список має вказівник лише на наступну ланку списку.

4. У чому особливість кільцевого списку?

Вказівник останньої ланки ссилається не в Nil, а на головну ланку списку.

5. Як видалити елемент у двонаправленому списку?

Присвоїти вказівник на наступну ланку тимчасовій змінній. Після цього присвоїти вже вказівник на наступну ланку цієї змінною вказівнику на наступну ланку початкової ланки.

6. Чим відрізняється стек від черги?

В стек можна тільки добавляти елементи, а в черзі - вставляти. Для обох доступні операції видалення та вибирання.

Висновок:

На цій лабораторній роботі я ознайомився із особливостями застосування динамічних об'єктів складної структури: списками, стеками та чергами; з операціями, які виконуються над елементами цих об'єктів; набув практичних навичок програмування з використанням динамічних об'єктів складної структури.