НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Обчислювальної техніки та програмування»

**Звіт з практичної роботи №6**

Тема: «Подання рядків у пам'яті»

Виконав:

Єрмаков М.В.

КІТ-120в

Дата: 14.04.21

Харків 2021

**Варіант 5**

**Хід роботи**

**На оцінку відмінно:**

Індивідуальне завдання: написати програму, в якій передбачити виконання вказаної операції над рядками за умови подання рядків у пам’яті двома способами. Порівняти подання рядків вказаними способами за такими показниками:

− обсягом використовуваної пам’яті;

− часом виконання функції.

Операцію та способи подання рядків вибрати з таблиці 6.1.

**Код:**

/\*\*

\* @author Ermakov M.

\* @date 14.04.2021

\* @brief EXERCISE №6

\*/

#include <iostream>

#include <vector>

using namespace std;

typedef char Data;

typedef struct Node Node;

struct Node

{

Data data;

Node\* next;

};

void addObject(Node\*\* plist, Data d);

Data delObject(Node\*\* plist);

void show(Node \* list);

void vectorReg(int F);

void blockOutput(int F);

int main() {

setlocale(LC\_ALL, "rus");

char choose;

cout << "Enter if vector - 'v', if block - 'b': ";

cin >> choose;

if (choose == 'v')

{

//регулируем длину вектора

int F;

cout << "Vector length: " << endl;

cin >> F;

//вывод вектора с регулированной длиной

vectorReg(F);

}

else if (choose == 'b')

{

//регулируем длину списка

int F;

cout << "Block length: " << endl;

cin >> F;

//вывод списка

blockOutput(F);

}

else

{

cout << "Invalid entered value. Enter if vector - 'v', if block - 'b'";

}

return 0;

}

void vectorReg(int F) {

char\* arrayInt = new char[F];

vector <char> s; //вектор размером на F элементов

for (int i = 0; i < F; ++i) //заполнение массива из F элементов числом 1

{

cin >> arrayInt[i];

}

for (int i = 0; i < F; i++)

{

s.push\_back(arrayInt[i]);

}

cout << "Vector contains: " << ends;

for (int i = 0; i < F; ++i)

cout << s[i] << ends; //вывод на экран

cout << endl;

vector <char> str;

//с какого по какой символ вектора выводим

int m;

int n;

cout << "Begin: " << ends;

cin >> m;

cout << "How many: " << ends;

cin >> n;

if ((m + n) < F) {

for (int i = m; i < n + m; i++)

{

str.push\_back(arrayInt[i]);

}

cout << "Finish vector contains: ";

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

cout << str[i] << ends;//вывод на экран

cout << endl;

}

else

{

cout << "Segmentation fault! The summ of begin and quantity had to be not less and not equal to summ of begging vector" << endl;

}

}

void blockOutput(int F) {

char\* arrayInt = new char[F];

for (int i = 0; i < F; i++) //заполнение массива из F элементов числом 1

{

cin >> arrayInt[i];

}

Node\* list = NULL;

for (int i = F-1; i >= 0; --i) //заполнение массива из F элементов числом 1

{

addObject(&list, arrayInt[i]);

}

show(list);

while (list != NULL)

{

delObject(&list);

}

//с какого по какой символ вектора выводим

int m;

int n;

cout << "Begin: " << ends;

cin >> m;

cout << "How many: " << ends;

cin >> n;

if ((m + n) < F) {

for (int i = n + m-1; i >= m; i--)

{

addObject(&list, arrayInt[i]);

}

show(list);

}

else

{

cout << "Segmentation fault! The summ of begin and quantity had to be not less and not equal to summ of begging vector" << endl;

}

}

void addObject(Node\*\* plist, Data d) {

Node\* p = new Node[sizeof(Node)];

p->data = d;

p->next = \*plist;

\*plist = p;

}

void show(Node\* list) {

printf("List: ");

for (Node \* p = list; p != NULL; p = p->next)

{

printf("%c", p->data);

}

printf("\n");

}

Data delObject(Node\*\* plist) {

if (\*plist == NULL) {

return 0;

}

else {

Node\* p = \*plist;

Data res = p->data;

\*plist = p->next;

free(p);

return res;

}

}

**Приклад результату виводу:**



Рисунок 1 – Результат виводу програми

**Принцип дії програми:**

Перед початком програми створюємо структуру списку з початковим та кінцевим показчиком й зі стрічковим значенням. Починаємо виконання програми. Спочатку оголошуємо змінну типу 'char' для визначення подальшого виводу програми: якщо 'v', то виводимо вектор, якщо 'b', то блок. Потім вводимо елементи масиву. Виводимо їх як вектор або блок. Потім очищуємо вектор або блок. Вводимо початок копіювання рядку, потім кількість символів. Переносимо в вектор або блок. Виводимо результат.



Блок-схема 1 – принцип роботи програми. Головна функція.



Блок-схема 2 – принцип роботи програми. Функція addObject().



Блок-схема 3 – принцип роботи програми. Функція delObject().



Блок-схема 4 – принцип роботи програми. Функція vectorReg().



Блок-схема 5 – принцип роботи програми. Функція blockOutput().



Блок-схема 6 – принцип роботи програми. Функція show().

**Висновок:** я отримав практичні навички та закріпив знання про можливі подання даних типу рядків та про операції над рядками.