Министерство образования Республики Беларусь

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**Лабораторная работа №3**

**По дисциплине «Основы алгоритмизации и программирования»**

**Тема: «Динамическая память»**

Выполнил: Шиковец Е.А.

Группа 21-ИТ-1

Проверила: Пантелейко А.Ф

Преподаватель-стажер кафедры ТП

Полоцк 2022

**Цель работы:** изучить принципы работы с памятью и функции для работы с ней. Научиться разрабатывать программы с использованием динамических структур на смежной памяти.

**Вариант № 4**

**Задание 1.** Создать указатель на переменную типа (а), выделить для нее динамическую память, считать ее значение и вывести на экран. Освободить память.

**Задание 2.** Создать указатель на строку длиной N символов. Ввести строку с клавиатуры. Уменьшить длину строки на 5 символов в конце добавить символ '\0'. Вывести результат на экран. Увеличить длину строки на M символов. Заполнить последние M элементов строки символом '\*'. Вывести результат на экран. Очистить память.

**Задание 3.** Создать массив типа (б), выделить память на N элементов. Считать элементы с клавиатуры и вывести полученный массив на экран. Очистить память.

**\* Задание 4.** Создать свободный массив типа (а). Считать количество элементов в строке и значения элементов с клавиатуры. Вывести результат на экран. Очистить память.

**\*\*Задание 5.** В динамический свободный массив записывать строки, заранее считанные с клавиатуры. То есть, сначала считывается строка, затем определяется ее длина, после чего она заносится в строку динамического массива. Вывести содержимое массива в виде текста. Очистить память.

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

int main()

{

double \*p;

p = (double \*)malloc(1000 \* sizeof(int));

cout << \*p;

free(p);

return 0;

}

**Пояснение к коду !**

Создается указатель типа double \*p. Далее выделяется для него динамическая память и считывается ее значение, значение выводится на экран. В конце динамическая память освобождается: free(p).



#include <iostream>

#include <cstring>

using namespace std;

int main()

{

int size = 0;

cout << "N" << endl;

cin >> size; // вводим размер строки

char \*str = new char[size]; // создаем указатель на строку длинной size

cin >> str;

str[strlen(str) - 5] = '\0'; // удаляем 5 последних букв

cout << str << endl; // выводим

int m = 0;

cout << "M" << endl;

cin >> m; // вводим m

for (int i = 0; i < m; i++) // создаем цикл

{

strcat(str, "\*"); // добавляем \* к строке

}

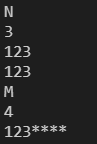
cout << str << endl; // выводим строку

free(str); // освобождаем память

return 0;

}

Создается указатель на строку длиной N символов. Строка и ее длинна вводится пользователем. Далее создается указатель на строку. Далее уменьшается длина строки на 5 символов, в конце добавляется символ '\0'. Далее длина строки увеличивается на M символов и заполняется символами'\*'. В конце результат выводится на экран, память очищается.



#include <iostream>

using namespace std;

int main()

{

int \*a; // создаем указатель

int n = 0; // создаем переменную типа int

cout << "Array size: " << endl;

cin >> n; // вводим n

a = (int \*) malloc(n \* sizeof(int)); // выделяем память для int

cout << "Array elements input: " << endl;

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

cin >> a[i]; // вводим элементы массива

cout << "Array elements output: " << endl;

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

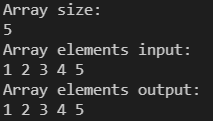
cout << a[i] << " "; // выводим элементы массива

free(a); // освобождаем память

return 0;

}

Создается указатель \*a, выделяется память для него, также вводятся необходимые переменные. Через цикл for вводятся элементы массива. Далее элементы массива выводятся и память, выделенная под массив освобождается.



#include <iostream>

using namespace std;

int main()

{

double \*\*a; // создаем указатель на указатели

int \*m; // создаем указатель на размеры

int n = 0;

cout << "Input number of lines: " << endl;

cin >> n;

a = (double \*\*) malloc(n \* sizeof(double \*));

m = (int \*) malloc(n \* sizeof(int)); // выделяем память

// Ввод элементов массива

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

{

cout << "Input number of columns for the " << i+1 << " line: ";

cin >> m[i]; // вводим элемент массива m

a[i] = (double \*) malloc(m[i] \* sizeof(double)); // выделяем память под него

for (int j = 0; j < m[i]; j++) cin >> a[i][j]; // вводим элемент a[i][j]

}

// Вывод элементов массива

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

{

cout << "Line " << i+1 << ": ";

for (int j = 0; j < m[i]; j++)

{

cout << a[i][j] << " \n"; // выводим элемент a[i][j]

}

}

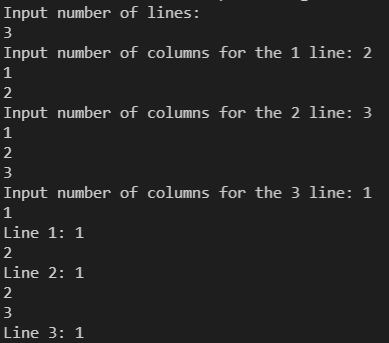
free(a); // освобождаем память

free(m);

return 0;

}

Создается указатель на указатели \*а и указатель на размеры \*m. Далее вводится кол-во строк, выделяется динамическая память для a и m, через цикл for происходит ввод элементов массива и выделение памяти для этих элементов. Далее через второй цикл for происходит вывод элементов массива. Далее память освобождается.



#include <iostream>

#include <cstring>

using namespace std;

void free(void \*ptr);

int main()

{

int n = 0;

cout << "Number of words: " << endl;

cin >> n;

string \*a = new string[n]; // создаем указатель на строку

for (int i = 0; i < n; i++) // Ввод элементов массива

{

string str;

cin >> str; // вводим строку

cout << "length -> " << str.length() << endl;

a[i] = str; // присваиваем значение

}

a[0][0] = toupper(a[0][0]); // делаем 1 букву заглавной

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

{

cout << a[i];

if(i != n-1) cout << " "; // добавляем пробелы между словами

}

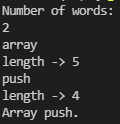
cout << ".";

free(a); // освобождаем память

return 0;

}

Создается указатель на строку \*а. Далее происходит ввод элементов массива через цикл for. В этом же цикле выводится длинна слов. Во втором цикле for выводится форматированный текст. Далее память освобождается.



**Вывод:**

Изучены принципы работы с памятью и функции для работы с ней. Разрабатывать программы с использованием динамических структур на смежной памяти.