**21.02.2024. Динамические переменные и одномерные массивы**

**Динамические переменные. Динамические одномерные массивы – создание/удаление, доступ к элементам, инициализация. Передача одномерного динамического массива в функцию. Примеры программ.**

**Динамические переменные**

**Динамическое выделение памяти.** Это способ запроса памяти у операционной системы запущенными программами по мере необходимости.

Динамическая память выделяется из области памяти, управляемой операционной системой — так называемой **кучи**. На современных компьютерах размер кучи может составлять **гигабайты** памяти.

Динамическоевыделение памяти — это способ выделения оперативной памяти программе во время её выполнения.

Динамические переменные находятся в памяти, выделенной для них, до тех пор, пока не будут удалены ***явным*** образом или пока программа *не завершит своё выполнение*.

Для управления динамическими переменными в языке программирования **С++** применяются две операции– **new** и **delete.**

Операция **new** запрашивает память у операционной системы и возвращает адрес начала выделенной области памяти. Для доступа к выделенной памяти следует создать указатель, значением которого будет данный адрес:

int \* ptr; ptr = new int; **или**

int \* ptr = new int; // выделение памяти для типа **int**

Операция **delete** освобождает выделенную память и возвращает её операционной системе:

delete ptr; // освобождение памяти

При создании динамической переменной допускается сразу её ***инициализация***:

int \* ptr = new int(5);

К указателю **ptr**можно применить *операцию разыменования* **(\*),** т.е. взять значение динамической переменной по её адресу:

cout<<\*ptr<<endl; // **5**

Динамические переменные **не имеют** прямого имени (**прямой адресации**). Для доступа к значениям динамических переменных используются только указатели (**косвенная адресация**), т.е. в этом случае указателистановятся ***единственным*** способом доступа к динамическим областям памяти (ячейкам).

// **Пример 1.** Использование динамических переменных в программе.

#include <iostream>

using namespace std;

int **main()**{

int val1 = 10, val2 = 20;

int\* ptr = new int(val1);

cout << "val1 = " << val1 << endl; // **10**

cout << "\*ptr = " << \*ptr << endl; // **10**

val1 = 5;

cout << "val1 = " << val1 << endl; // **5**

cout << "\*ptr = " << \*ptr << endl; // **10**

\*ptr = 7;

cout << "val1 = " << val1 << endl; // **5**

cout << "\*ptr = " << \*ptr << endl; // **7**

delete ptr; // **память надо освободить**

ptr = new int(val2);

cout << "val2 = " << val2 << endl; // **20**

cout << "\*ptr = " << \*ptr << endl; // **20**

delete ptr; // **память надо освободить**

}

**Динамические одномерные массивы**

В языке программирования **С++** для создания/удаления ***динамического одномерного массива*** также используются операции **new** и **delete:**

int n; cin>>n; // например, введём 4

int \* ptr = new int[n]; // создание динамического массива

delete [] ptr; // освобождение памяти

**Доступ к элементам массива*.*** После создания динамического массива можно с ним работать по полученному указателю.

Доступк **i**-ому элементу одномерного динамического массива, адрес начала которого хранится в указателе **ptr,** возможен следующими способами:

ptr[i] \*(ptr + i) // значение **i-го** элемента массива

Соответственно их адреса:

&ptr[i] (ptr + i) // адрес **i-го** элемента массива

// **Пример 2.** Создание/удаление **динамического** одномерного массива.

// Размер массива вводится.

#include <iostream>

using namespace std;

int **main()**{

int n;

cout << "Enter size array: "; cin >> n; // **5**

int \* ptr = new int[n]; // выделение памяти

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

ptr[i] = i + 1;

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

cout << \*(ptr + i) << ' '; // **1 2 3 4 5**

cout << endl;

for (int \* q = ptr; q != ptr + n; q++) // вывод массива

cout << \*q << ' '; // **1 2 3 4 5**

cout<<endl;

delete [] ptr; // освобождение памяти

}

**Инициализация массива:**

int n; cin>>n; // например, введём **4**

int \* ptr = new int[n](); // все элементs равны **нулю**

int \* ptr = new int[n]{}; // все элемент равны **нулю**

int \* ptr = new int[n]{ 1, 2, 3, 4 }; // 1 2 3 4

int \* ptr = new int[n]{ 1, 2 }; // 1 2 0 0

int \* ptr = new int[n]{ 1, 2, 3, 4, 5, 6 }; // **нет ошибки** 1 2 3 4

int \* ptr = new int[**4**]{ 1, 2, 3, 4, 5, 6 }; // **ошибка** -- **много инициализаторов**!!

**Передача одномерного динамического массива в функцию**

// **Пример 3.** Найти сумму элементов одномерного динамического

// массива. Массив создать в функции **main()**.

// Составить и использовать в программе функции **создания/вывода** массива,

// **нахождения суммы** элементов массива, передавая им адрес массива.

#include <iostream>

using namespace std;

void **init**(int\*, int); // прототип  **init**()

void **print**(int\*, int); // прототип  **print**()

int **sum**(int\*, int); // прототип  **sum()**

int **main()**{

int n;

cout << "Enter size array: ";cin >> n; // **5**

int\* ptr = new int[n]; // создание массива

**init**(ptr, n); // вызов **init**()

cout << "Array : " << endl;

**print**(ptr, n); // **6 -9 0 -7 7**

int summa = **sum**(ptr, n);

cout << "Summa = " << summa << endl; // **Summa** = **-3**

delete [] ptr;

}

void **init**(int\* p, int n) { // функция **init()**

srand(n);

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

p[i] = rand() % 15 - rand() % 10;

}

void **print**(int\* p, int n) { // функция **print()**

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

cout << p[i] << ' '; // или cout<<\*(p+i)<<' ';

cout << endl;

}

int **sum**(int\* p, int n) { // функция **sum()**

int summa = 0;

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

summa = summa + p[i];

return summa;

}

// **Пример 4.** Составить и использовать в программе функции

// **создания/вывода/удаления** **динамического** одномерного массива.

// Для выбора функций использовать **меню.**

**//** Память для массива выделить в функции создания и возвратить адрес

// массива в **main().** Предусмотреть возможность **изменять** размер

// массива во время выполнения программы.

#include <iostream>

using namespace std;

int\* **create**(int &n); // прототип **create**()

void **print**(int\*, int); // прототип **print()**

void **delMemory**(int\*); // прототип **delMemory()**

int **main()**{

int n, select,\*ptr = 0;;

bool flag = **true**;

**while(**flag**)**{

cout<<"Enter the label number:\n";

cout << "create----->1" << " print------>2" << '\n'

<< "delete----->3" << " exit------->4" << endl;

cin>>select; // выбор функции

**switch**(select){

case 1: ptr = **create**(n); break;

case 2: print(ptr, n); break;

case 3: delete(ptr); break;

case 4: cout<<"End!"; flag = **false**;

}

}

cout<<endl;

}

int\* **create**(int &n){ // функция **create()**

cout<<"Enter size array: "; cin>>n; // ввод размера массива

int \* p = new int[n];

srand(n);

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

p[i] = rand() % 15 - rand() % 10;

**return** p;

}

void **print**(int \* p, int n){ // функция **print()**

cout<<"**Array**: "<<endl;

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

cout<<\*(p + i )<<' '; // cout<<p[i]<<' ';

cout<<endl;

}

void **delMemory**(int \* p){ // функция **delMemory()**

delete [] p;

}

Не забывать **освобождать память** перед вводом нового размера массива и перед выходом из программы**!!!**

**Результат:**

1. Enter the label number: **4.** Enter the label number:

create----->1 print------>2 create----->1 print------>2

del-------->3 exit------->4 del-------->3 exit------->4

**1 1**

Enter size array: **5** Enter size array: **7**

1. Enter the label number: **5.** Enter the label number:

create----->1 print------>2 create----->1 print------>2

del-------->3 exit------->4 del-------->3 exit------->4

**2 2**

**Array: Array:**

6 -9 0 -7 7 -1 5 5 -2 3 -6 6

1. Enter the label number: **6.** Enter the label number:

create----->1 print------>2 create----->1 print------>2

del-------->3 exit------->4 del-------->3 exit------->4

**3 3**

**7.** Enter the label number:

create----->1 print------>2

del-------->3 exit------->4

**4**

**End!**

// **Пример 5.** Найти сумму элементов одномерного динамического массива.

// Создать и использовать функции для **выделения/освобождения памяти,**

**// создания (заполнения), вывода массива, нахождения суммы элементов.**

**// массива.** А также функцию **выбора типа** заполнения массива – через ввод

// значений элементов массива или случайными числами.

#include <iostream>

#include <ctime>

using namespace std;

int\* **memArray**(int); // выделение памяти

int **choiceVvod()**; // выбор типа заполнения массива

void **fillConsol**(int\*, int); // заполнение массива **с консоли**

void **fillRand**(int\*, int); // заполнение **rand**-числами до **100**

void **printArray**(int\*, int ); // вывод массива

int **sum**(int\*, int); // сумма элементов массива

void **delMemory**(int\*p); // освобождение памяти

int **main()**{

const int maxSIZE = 40; // максимальный размер массива

srand(time(NULL));

int select = choiceVvod(); // выбор создания массива

cout << "select = " << select << endl;

int size = 0; // размер массива

while (size < 2 || size > maxSIZE) {

cout << "Enter size of array (from 2 to " << maxSIZE << ") ";

cin >> size;

}

cout << "size = " << size << endl;

int \* ptr = **memArray**(size); // вызов **memArray()**

switch (select) {

case 1: **fillConsol**(ptr, size); break; // вызов **fillConsol()**

case 2: **fillRand**(ptr, size); break; // вызов **fillRand()**

default: break;

}

**printArray**(ptr, size); // вызов **printArray()**

int summa = **sum**(ptr, size); // вызов **sum()**

cout << "Sum of elements in array = " << summa << endl;

**delMemory(**ptr**);** // вызов **delMemory()**

}

int\* **memArray**(int size) { // функция **memArray()**

int\* p = new int[size];

if (!p) { cout << "Error! No memory\n"; system("pause"); exit(1); }

return p;

}

int **choiceVvod(){** // функция **choiceVvod()**

int choice = 0;

while (choice != 1 && choice != 2) {

cout << "1: array from console \n"

<< "2: array with random numbers \n";

cin >> choice;

}

return choice;

}

void **fillConsol**(int \* p, int size) { // функция **fillConsol()**

cout << "Input array"<<endl;

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

cin >> p[i];

}

void **fillRand**(int \* p, int size) { // функция **fillRand()**

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

p[i] = rand() % 100;

}

void **printArray**(int\* p, int size) { // функция **printArray()**

for (int\* q = p; q != p + size; q++)

cout << \*q << ' ';

cout << endl;

}

int **sum(**int \* p, int size) { // функция **sum()**

int sum = 0;

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

sum = sum + p[i];

return sum;

}

void **delMemory**(int \* p){ // функция  **delMemory()**

delete [] p;

}

**Результат:**

1: array from console

2: array with random numbers

2

select = 2

Enter size of array (from 2 to 40) **8**

size = 8

82 29 58 28 7 82 21 67 Sum of elements in array = 374

**Д. З. на 28 02.2024**

**Постановка задания.**

**!!!** В программах использовать **только** учебный материал .

!!! В тексте программы **обязательно** должны быть **условие задачи** и **тестовый пример**, использованный при отладке программы.

**Ограничения:**

* не создавать для хранения цифр числа **массив;**
* не задавать числа как **символьный массив**;
* не использовать объекты класса **string**;
* не использовать стандартные функции.

**Требования к программе:**

* в программе использовать ***динамические*** одномерные массивы;
* основные этапы работы с массивом – ввод/вывод массива, обработка элементов массива, всего массива оформить в виде функций;
* использовать **меню** (для задачи без звёздочки **по желанию**) для выбора функций при выполнении программы.
* результат выводить в доступном и понятном виде.

**Задачи**:

1. Ввести/вывести одномерный массив натуральных чисел размера ***n***. Если массив содержит хотя бы один **элемент-палиндром** и упорядочен по убыванию значений элементов, то присвоить переменной **pr** значение **1**, иначе – значение **0**.
2. \*Ввести/вывести одномерный массив целых чисел размера ***n***.Преобразовать исходный массив, добавив в цепочку максимальной длины идущих подряд одинаковых элементов ещё один элемент, равный сумме элементов данной цепочки. Дополнительный массив не использовать. Например:

исходный массив **mas ---** 5 5 5 2 3 1 1 1 1 6 1 4 4 4 7 8 9

массив-результат **mas ---** 5 5 5 2 3 1 1 1 1 4 6 1 4 4 7 8 9

1. \*\*Ввести/вывести одномерный массив натуральных чисел размера ***n***. Удалить в исходном массиве цепочку максимальной длины идущих подряд элементов, значения которой расположены по возрастанию (нули не учитывать). Дополнительный массив не использовать. Например:

исходный массив **mas** --- 2 3 0 5 0 6 1 2 3 4 0 5 6 7 2 0 4 1 5

цепочка **max** длины ----- 1 2 3 4 0 5 6 7

массив-результат **mas ---** 2 3 0 5 0 6 2 0 4 1 5