МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Програмування. Частина 1. Основи програмування

Лабораторна робота №6

Тема: Одновимірні масиви

Варіант № 10

Виконав:

студент групи ІС-34

Колосов Ігор

Перевірив:

Драган М.С.

Київ-2023

**Завдання №1**



Вирішення:

#include <iostream>

#include <algorithm>

#include <memory.h>

#include <vector>

int \*createArray(int \*len) {

std::cout << " > Enter length of array: ";

std::cin >> \*len;

int \*array = new int[\*len];

for (int i = 0; i < \*len; i++) {

std::cout << " > array [" << i << "]:";

std::cin >> array[i];

}

return array;

}

int \*printArray(int \*array, int len) {

std::cout << "Array: ";

for (int \*p = array; p < array+len; p++)

std::cout << \*p << " ";

std::cout << std::endl;

return array;

}

bool numInArray(int num, int \*array, int len) {

for (int \*p = array, k = 0; p < array+len; p++)

if (\*p == num) return true;

return false;

}

bool numInArrayFaster(int num, int \*array, int len) {

#pragma omp parallel for

for (int \*i=array, \*j=array+2; i < array+len; i++, i++, j++, j++)

if (\*i == num || \*j == num) return true;

return false;

}

*/\* returns output array with unique numbers from array*

*\* modifies entriesLen to be the length of the output array \*/*

int \*unique(int \*array, int len, int \*entriesLen) {

int \*output = new int[len];

memset(output, 0, len\*sizeof(int));

\*entriesLen = 0;

for (int i = 0; i < len; i++) {

if (!numInArray(array[i], array, i)) {

output[(\*entriesLen)++] = array[i];

}

}

return (int\*)realloc(output, \*entriesLen\*sizeof(int));

}

int main () {

int \*len = new int, \*entries = new int(0); *// Initialize entries to 0*

int \*array = createArray(len);

std::cout << " < array > " << std::endl;

printArray(array, \*len);

std::cout << " < unique > " << std::endl;

int \*uniqueArray = unique(array, \*len, entries);

printArray(uniqueArray, \*entries);

delete[] array;

delete len;

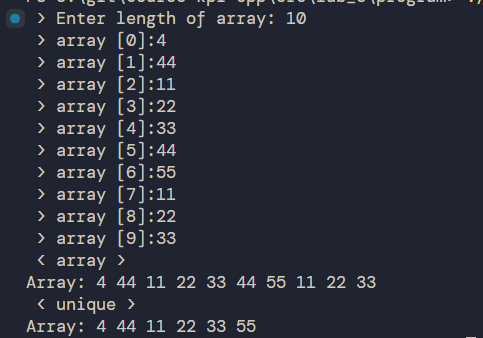
delete entries;

delete[] uniqueArray; *// Don't forget to delete uniqueArray*

return 0;

}

Вивід:



**Завдання №2**



Вирішення:

#include <iostream>

#include <algorithm>

#include <memory.h>

#include <vector>

#include <time.h>

struct ArrMinMax{

int\* min;

int\* max;

};

int \*printArray(int \*array, int len) {

std::cout << "Array: ";

for (int \*p = array; p < array+len; p++)

printf("%2d ", \*p);

std::cout << std::endl;

return array;

}

int\* randomArray(int \*array, int len) {

for (int \*p = array; p < array+len; p++)

\*p = rand() % 100;

return array;

}

ArrMinMax minmax(int \*array, int\* end) {

int \*max = nullptr, \*min = nullptr;

int \*p = array;

while (p++ < end-1) {

if (!min || \*p <= \*min) min = p;

if (!max || \*p >= \*max) max = p;

}

std:: cout << \*min << " " << \*p << " " << \*max << '\n';

return {min, max};

}

int main () {

srand(time(NULL));

int len = 10;

ArrMinMax valuesToSwap;

int \*array = randomArray(new int[len], len);

valuesToSwap = minmax(array, array+len);

std::cout << " < minmax > " << std::endl;

if (valuesToSwap.min != nullptr && valuesToSwap.max != nullptr){

printf(" - indicies: [%2d] <-> [%2d]\n", array + len - valuesToSwap.min, array+ len - valuesToSwap.max);

printf(" - values : [%2d] <-> [%2d]\n", \*valuesToSwap.min, \*valuesToSwap.max);

printArray(array, len);

std::swap(\*valuesToSwap.min, \*valuesToSwap.max);

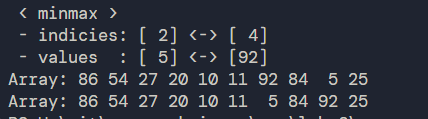
printArray(array, len);

}

return 0;

}

Вивід:



**Завдання №3**



Вирішення:

#include <iostream>

void swap(int \*array, int a, int b) {

int temp = array[a];

array[a] = array[b];

array[b] = temp;

}

int partition(int \*array, int begin, int end) {

int pivot = array[end];

int i = begin - 1;

for (int j = begin; j < end; j++)

if (array[j] <= pivot)

swap(array, j, ++i);

swap(array, end, ++i);

return i;

}

int\* quick\_sort(int \*array, int begin, int end) {

if (begin < end) {

int i = partition(array, begin, end);

quick\_sort(array, begin, i-1);

quick\_sort(array, i+1, end);

}

return array;

}

int \*createArray(int \*len) {

std::cout << " > Enter length of array: ";

std::cin >> \*len;

int \*array = new int[\*len];

for (int i = 0; i < \*len; i++) {

std::cout << " > array [" << i << "]:";

std::cin >> array[i];

}

return array;

}

int \*printArray(int \*array, int len) {

std::cout << "Array: ";

for (int \*p = array; p < array+len; p++)

printf("%2d ", \*p);

std::cout << std::endl;

return array;

}

int main () {

int \*len = new int;

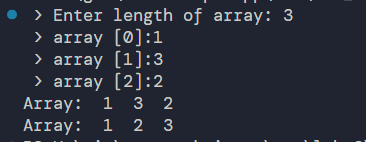
int \*array = createArray(len);

delete printArray(quick\_sort(printArray(array, \*len), 0, \*len), \*len);

return 0;

}

Вивід:



**Контрольні запитання:**

1. Дати означення масиву та типу масиву
   1. Масив – це розташовані поруч комірки в пам’яті що зберігають інформацію послідовно
   2. Це структурований тип даних, множина допустимих значень котрого складається з усіх масивів, для яких зафіксовано
2. Якими є властивості масивів даних?
   1. однорідність — усі елементи належать одному типу;
   2. сталість — вимірність масиву задається під час його оголошення і не змінюється протягом роботи з ним;
   3. рівнодоступність — спосіб доступу до всіх елементів є однаковим;
   4. послідовність розташування — усі елементи масиву розташовані в послідовних комірках оперативної пам’яті;
   5. індексованість — елементи однозначно ідентифікуються своїми індексами;
   6. упорядкованість індексу — індексний тип має бути простим порядковим типом даних.
3. Яким є принцип зображення масиву в оперативній пам’яті?
   1. -
4. Як здійснюється доступ до елементів одновимірного масиву?
   1. <ім’я масиву>[<індекс масиву>]
   2. Використовуючи показники
5. Чи може список ініціалізації масиву містити більше (менше) значень, ніж вказано в оголошенні масиву?
   1. Може мати менше, але не може мати більше.
6. Які базові операції обробки одновимірних масивів?
   1. введення та виведення масиву;
   2. ініціалізація масиву;
   3. копіювання масиву;
   4. пошук максимального або мінімального елемента;
   5. обчислення узагальнювальних характеристик (сум елементів, їх добутків);
   6. пошук заданого елемента;
   7. перестановка елементів або обмін значеннями між елементами масиву;
   8. вставка та видалення елемента.
7. Чим відрізняються методи сортування масивів?
   1. Просторова складність та часова складність
8. Приведіть найшвидший та найповільніший алгоритми сортування масивів
   1. Богосорт - є одночасно найповільнішим та найшвидшим алгоритмом
   2. Найшвидшим можна ще вважати модифіцації квік-сорта
9. Чим відрізняються алгоритми пошуку в упорядкованому та неупорядкованому масивах?
   1. Відрізняється метод пошуку, в упорядкованому він буде більше, а також ми можемо використовувати подвійні пошук
10. Як повернути масив із функції?
    1. Використовуючи тип функції з поінтер
11. Що означає подвійний покажчик на одновимірний масив?
    1. Створення масиву вказівників

**Висновок:**

Ознайомився з особливостями типу масиву, опанував технологію застосування масиву даних, навчився розробляти алгоритми та програми із застосування одновимірних масивів.