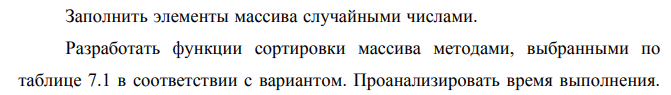
**Лабораторна робота №7**

Розробка та реалізація програм сортування масивів

**Мета роботи**: набуття практичних навичок розробки та реалізації програм сортування масивів.

*Завдання*





**Хід роботи**

1. **Текст програми**

#include <iostream>

#include <fstream>

#include <algorithm>

#include <vector>

#include <string>

#include <ctime>

void selectionSort(std::vector<int>& arr);

void quickSort(std::vector<int>& arr, int l, int r);

void printArr(std::vector<int>& arr, int maxElementsToPrint = 500);

void writeVector(std::vector<int>& arr);

std::vector<int> getSizesToTest();

void printHello();

bool askIfExit();

void runTests(std::vector<int> &sizesToTest);

void test(int size);

template<typename F>

long long measure(F&& f);

int main() {

srand((int)time(0));

printHello();

while (true) {

auto sizesToTest = getSizesToTest();

runTests(sizesToTest);

if (askIfExit()) break;

std::cout << "\n- - - - - - - -\n\n";

}

return 0;

}

void runTests(std::vector<int>& sizesToTest) {

if (sizesToTest.empty()) {

std::cout << "No sizes to test\n";

}

std::cout << std::endl;

for (auto& size : sizesToTest) {

test(size);

}

}

void test(int size) {

std::cout << "Testing for " << size << " elements:\n";

std::vector<int> arr(size);

writeVector(arr);

// Quick Sort

std::cout << "Quick Sort: ";

std::vector<int> arrToQuickSort(arr);

auto timeToQuickSort = measure([&arr, &arrToQuickSort] {

arrToQuickSort = std::vector<int>(arr);

quickSort(arrToQuickSort, 0, arrToQuickSort.size() - 1);

});

std::cout << timeToQuickSort << std::endl;

// Selection Sort

std::cout << "Selection Sort: ";

std::vector<int> arrToSelectionSort;

auto timeToSelectionSort = measure([&arr, &arrToSelectionSort] {

arrToSelectionSort = std::vector<int>(arr);

selectionSort(arrToSelectionSort);

});

std::cout << timeToSelectionSort << std::endl;

std::cout << "- - - - -\n";

}

std::vector<int> getSizesToTest() {

std::vector<int> sizes;

std::cout << "Enter sizes to test (non-positive to complete):\n";

while (true) {

int size = prompt<int>("> ");

if (size <= 0) break;

sizes.push\_back(size);

}

return sizes;

}

void quickSort(std::vector<int>& arr, int l, int r) {

int i = l, j = r;

int pivot = arr[(i + j) / 2];

while (i < j) {

while (arr[i] < pivot) i++;

while (arr[j] > pivot) j--;

if (i > j) break;

std::swap(arr[i], arr[j]);

i++; j--;

}

if (i < r) quickSort(arr, i, r);

if (j > l) quickSort(arr, l, j);

}

void selectionSort(std::vector<int>& arr) {

int size = arr.size();

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

int min = arr[i];

int minIndex = i;

for (int j = i + 1; j < size; j++) {

if (arr[j] >= min) continue;

min = arr[j];

minIndex = j;

}

std::swap(arr[i], arr[minIndex]);

}

}

void writeVector(std::vector<int>& arr) {

for (auto &item : arr) {

item = (rand() % 89) + 10;

}

}

void printArr(std::vector<int>& arr, int maxElementsToPrint) {

int size = std::min((int)arr.size(), maxElementsToPrint);

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

std::cout << arr[i] << " ";

}

if ((int)arr.size() > size) {

std::cout << "(...)";

}

std::cout << std::endl;

}

void printHello() {

std::cout << "\* \* \* Algorithms: Sorting Methods \* \* \*\n\n";

}

bool askIfExit() {

return (prompt<std::string>("\nRepeat? (0 if not): ") == "0");

}

template<typename F>

long long measure(F&& f) {

auto start = std::clock();

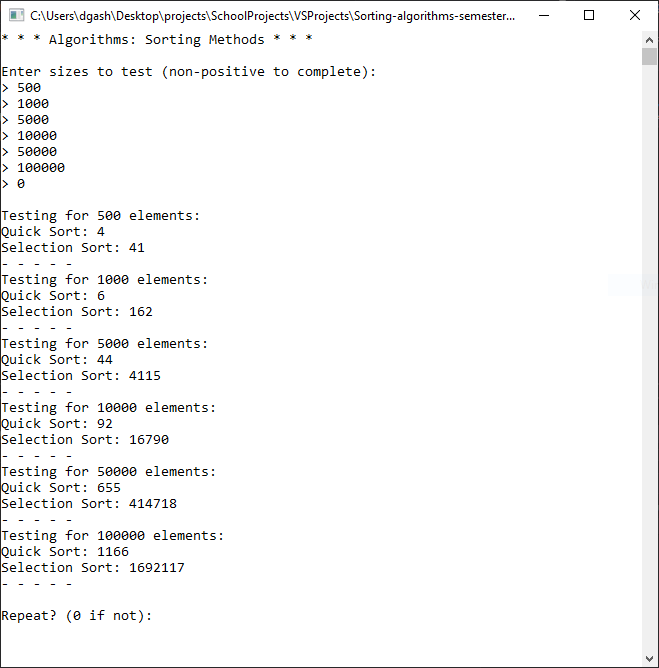
f();

auto time = std::clock() - start;

return (long long)time;

}

1. **Результат виконання програми**

****

**Висновок:** на цій лабораторній роботі я набув практичні навички розробки та реалізації програм сортування масивів.