**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Algorytmy i struktury danych | **Zadanie** | 9\_02 |
| **Autor** | Kamil Borkowski | **Grupa** | WCY22IY1S1 |
| **Temat** | Library Sort | | |

1. Treść

Zaimplementuj algorytm sortowania metodą biblioteczną (ang. Library Sort).

* 1. Metoda realizacji

Program wykorzystuje prosty interfejs użytkownika do wyboru operacji, wczytywania danych i sortowania za pomocą algorytmu Library Sort.

* 1. Założenia / ograniczenia dotyczące danych:
     1. Dane wejściowe *(opis)*

Wartości do sortowania – wczytywana z pliku/klawiatury/generatora

* + 1. Dane wyjściowe *(opis)*

Wartości posortowane – wyświetlone na ekranie.

1. Realizacja
   1. Algorytm *(lista kroków lub schemat blokowy)*
2. Tworzymy dynamicznie alokowaną tablicę pomocniczą o nazwie shelves. Tablica ta będzie przechowywała elementy w trakcie sortowania.
3. Szukamy miejsca na "półce" (shelves), porównując wartość current z elementami już umieszczonymi na półce.
4. Jeśli znajdziemy miejsce na półce, przesuwamy elementy na półce w prawo, aby zrobić miejsce dla current.
5. Dodajemy puste miejsce na półce, inkrementując shelvesSize.
6. Umieszczamy current na półce na odpowiednim miejscu.
7. Przepisujemy posortowane elementy z półki z powrotem do oryginalnej tablicy.
8. Zwalniamy pamięć zaalokowaną dynamicznie dla tablicy pomocniczej shelves.

Kod źródłowy

//================================================

//Zadanie 9\_02 jezyk C++

//Library Sort

//WCY22IY1S1 Borkowski Kamil

//================================================

#include <iostream>

#include <ctime>

#include <cstdlib>

#include <cstdio>

#include <fstream>

using namespace std;

//funkcja przesuwajaca elementy tablicy

void shiftElements(int arr[], int start, int end) {

for (int i = end; i > start; --i) {//przesuwanie od start do end wlacznie

arr[i] = arr[i - 1];

}

}

void librarySort(int arr[], int n) {

int\* shelves = (int\*)malloc(n \* sizeof(int));

int shelvesSize = 0;

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

int current = arr[i];

int j = 0;//szukanie miejsca na polce dla elementu

while (j < shelvesSize && shelves[j] <= current) {

++j;

}

shiftElements(shelves, j, shelvesSize);//dodawanie pustych miejsc

shelvesSize++;

shelves[j] = current;//wstawianie elementu na polce

}

//przepisanie posortowanych elementow z polki do oryginalnej tablicy

int sortedIndex = 0;

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

if (shelves[i] == 0) {//obsluga pustych miejsc

sortedIndex++;

} else {

arr[sortedIndex++] = shelves[i];//wstawianie elementu z polki do tablicy

}

}

free(shelves);

}

int main()

{

srand((time(0)));

int arr[10000] = {};

int i,k=0,n,g,d,w;

int pom=1,z;//pom oblusguje interfejs, z= nr komendy z interfejsu

cout << "WCY22IY1S1 Borkowski Kamil" << endl;

cout << "Program sortowania Library Sort" << endl;

cout << "[0] - wyjscie z programu" << endl;

cout << "[1] - wczytaj dane z pliku(we.txt)" << endl;

cout << "[2] - wczytaj dane z generatora" << endl;

cout << "[3] - wczytaj dane z klawiatury" << endl;

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

while(pom){

cout << "Wprowadz nr operacji: ";

cin >> z;

switch (z){

case 0:

cout<<"Wychodzenie..."<<endl;

pom = 0;

break;

case 1:

{

k=0;

ifstream in;

in.open("we.txt");

int p;

i=0;

while(in>>p)

{

arr[i]=p;

i++;

}

n=i;

cout<<"Tablica zostala usupelniona danymi z pliku we.txt"<<endl<<endl;

cout <<"Przed sortowaniem metoda library sort: "<<endl;;

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

{

cout<<arr[i]<<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

librarySort(arr,n);

k=0;

cout<<endl<<"Po sortowaniu metoda library sort: "<<endl;;

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

cout << arr[i] <<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

cout<<endl;

in.close();

break;

}

case 2:

{

k=0;

cout<<"Podaj ilosc liczb do sortowania"<<endl;

cin>>n;

cout<<"Podaj zakres liczb do sortowania"<<endl;

cout<<"Granica dolna: ";

cin>>d;

cout<<"Granica gorna: ";

cin>>g;

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

arr[i]=(rand() % (g-d+1))+d; // liczby od 1 do 100

}

cout <<"Przed sortowaniem metoda library sort: "<<endl;;

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

{

cout<<arr[i]<<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

librarySort(arr,n);

k=0;

cout<<endl<<"Po sortowaniu metoda library sort: "<<endl;;

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

cout << arr[i] <<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

cout<<endl;

break;

}

case 3:

{

k=0;

cout<<"Podaj ilosc liczb do sortowania"<<endl;

cin>>n;

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

{

w=0;

cout<<"Podaj argument: ";

cin>>w;

arr[i]=w;

}

cout <<"Przed sortowaniem metoda library sort: "<<endl;;

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

{

cout<<arr[i]<<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

librarySort(arr,n);

k=0;

cout<<endl<<"Po sortowaniu metoda library sort: "<<endl;;

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

cout << arr[i] <<'\t';

k++;

if(k==10)

{

k=0;

cout<<endl;

}

}

cout<<endl;

break;

}

default:

{

cout<<"Podano bledny nr operacji"<<endl<<endl;

break;

}

}

}

return 0;

}

* 1. Dane wejściowe *(ekran uruchomienia programu)*

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* 1. Dane wyjściowe *(ekran uruchomienia programu)*

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, oprogramowanie

Opis wygenerowany automatycznie

1. Wnioski *(złożoność asymptotyczna algorytmu przy użyciu notacji* ***O lub innej i inne wnioski)***

Złożoność obliczeniowa algorytmu sortowania Library Sort to O(n^2), gdzie "n" to liczba elementów w tablicy.