
Алгоритми та структури даних. Основи алгоритмізації

Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 8 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження лінійних алгоритмів»

Варіант 9

Виконав студент Григоренко Родіон Ярославович
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська А.С.
(прізвище, ім'я, по батькові)

Київ 2021

Алгоритми та структури даних. Основи алгоритмізації

Лабораторна робота 8

Дослідження алгоритмів пошуку та сортування

Мета – дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Індивідуальне завдання

9	6 x 6	Цілий	Із додатних значень елементів головної діагоналі двовимірного масиву. Відсортувати методом Шела за зростанням.
---	-------	-------	--

Варіант 9 Постановка задачі

Результатом є виведення елементів впорядкованого методом Шела одновимірного масиву з додатних елементів головної діагоналі початкового двовимірного масиву.

Побудова математичної моделі

Складемо таблицю імен змінних та функцій

Змінна	Тип	Ім'я	Призначення
Двовимірний динамічний масив	Показчик на масив показчиків(Цілочисельний)	arr	Початкове дане
Динамічний одновимірний масив	Показчик(Цілочисельний)	A	Результат
Довжина масиву arr	Цілочисельний	N	Початкове дане
Довжина масиву A	Цілочисельний	n	Проміжні дані
Показчик на змінну n	Показчик	pn	Проміжні дані
Параметри арифметичних циклів	Цілочисельний	i,j	Проміжні дані
Крок алгоритму Шела	Цілочисельний	step	Проміжні дані
Функція для ініціювання початкового масиву	int**	fill	Ініціювання початкового масиву
Функція для ініціювання одновимірного	Int*	arr2	Ініціювання другого масиву

Алгоритми та структури даних. Основи алгоритмізації

динамічного масиву додатних елементів головної діагоналі початкового масиву			
Алгоритм Шела	void	shell	Сортування одновимірного масиву алгоритмом Шела
Функція для заповнення масиву випадковими значеннями	Цілочисельний	rand	Заповнення масиву випадковими значеннями

fill - функція для ініціювання початкового двовимірного динамічного масиву.

init - функція для ініціювання одновимірного масиву з додатних елементів головної діагоналі початкового двовимірного масиву.

shell - функція, що сортує масив A, за допомогою алгоритму Шела.

arr - початковий масив.

A - одновимірний масив з додатних елементів головної діагоналі початкового двовимірного масиву.

N, n - розмірності масивів.

Розв'язання

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо функцію fill.

Крок 3. Деталізуємо функцію init.

Крок 4. Деталізуємо функцію shell.

Псевдокод

Крок 4

функція fill(N)

int** arr = new int *[N]

для i від 0 до N, збільшувати на 1

arr[i] = new int[N]

для i від 0 до N, збільшувати на 1

для j від 0 до N, збільшувати на 1

arr[i][j] = rand() - 16000

повернути arr

все функція

функція init(**arr, N, *pn)

Алгоритми та структури даних. Основи алгоритмізації

```
int* A = new int
int n = 0
для і від 0 до N, збільшувати на 1
    для j від 0 до N, збільшувати на 1
        якщо i == j && arr[i][j] > 0
            то
                A[n] = arr[i][j]
                n++
*pn = n
повернути A
```

все функція

```
функція shell(*arr, N)
    int memory
    для step від N/2 до 1, зменшувати в два рази
        для i від step до N, збільшувати на 1
            для j = i при (j >= step && arr[j - step] > arr[j]) , зменшувати на step
                memory = arr[j]
                arr[j] = arr[j - step]
                arr[j - step] = memory
```

все функція

```
функція output(*A, N, **arr, n)
    для і від 0 до N, збільшувати на 1
        для j від 0 до N, збільшувати на 1
            виведення arr[i][j]
    для і від 0 до N, збільшувати на 1
        виведення A[i]
```

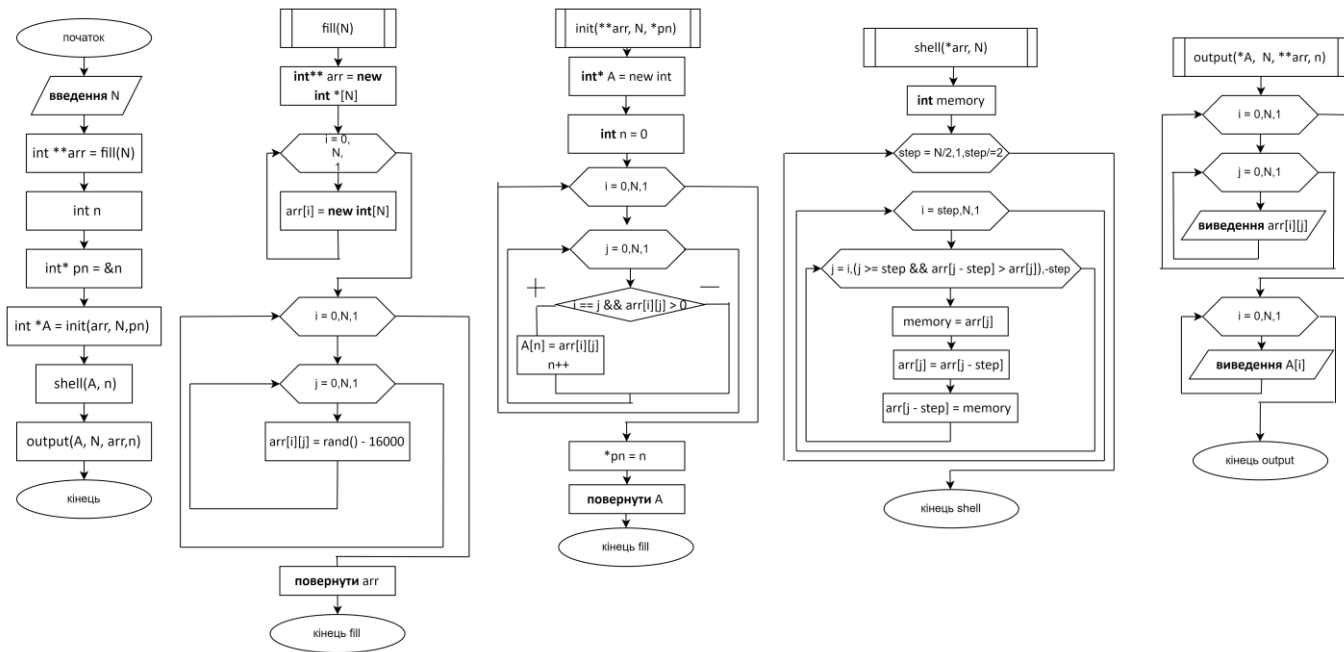
все функція

початок

```
int N
введення N
int **arr = fill(N)
int n
int* pn = &n
int *A = init(arr, N, pn)
shell(A, n)
output(A, N, arr, n)
```

кінець

Блок-схема Крок 4



Код

Алгоритми та структури даних. Основи алгоритмізації

```
#include <stdlib.h>
#include <iostream>
#include <time.h>
using namespace std;
void shell(int* arr, int n);
int* init(int** arr,int N,int *pn);
int** fill(int N);
void output(int* A, int N, int** arr, int n);
int main()
{
    int N;
    cout << "Array size: ";
    cin >> N;
    int **arr = fill(N);
    int n;
    int* pn = &n;
    int *A = init(arr, N,pn);
    shell(A, n);
    output(A, N, arr,n);
}
```

```
void shell(int* arr, int N)
{
    int memory;
    for (int step = N / 2; step >= 1; step /= 2) {
        for (int i = step; i < N; i++) {
            for (int j = i; j >= step && arr[j - step] > arr[j]; j -= step) {
                memory = arr[j];
                arr[j] = arr[j - step];
                arr[j - step] = memory;
            }
        }
    }
}

int** fill(int N) {
    srand(time(NULL));
    int** arr = new int *[N];
    for (int i = 0; i < N; i++) {
        arr[i] = new int[N];
    }
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            arr[i][j] = rand()-16000;
        }
    }
    return arr;
}
```

Алгоритми та структури даних. Основи алгоритмізації

```
int* init(int** arr,int N,int *pn) {
    int* A = new int;
    int n = 0;
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            if (i == j && arr[i][j] > 0) {
                A[n] = arr[i][j];
                n++;
            }
        }
    }
    *pn = n;
    return A;
}


void output(int* A, int N, int** arr, int n) {
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++) {
            cout << arr[i][j] << " ";
        }
        cout << '\n';
    }
    cout << '\n' << '\n';
    for (int i = 0; i < n; i++) {
        cout << A[i] << " ";
    }
    cout << '\n';
}
```

Результат коду

 C:\Users\STRIX\source\repos\Lab8_ASD\Debug\Lab8_ASD.exe

```
Array size: 6
1210  -8692  279  -11873  7930  -2490
-580  7414  -15006  1283  -1567  9396
-9752  -7306  11453  15038  16424  6870
7806  6946  1614  -1616  14406  -13307
5431  4993  5491  14813  11219  -8572
-1501  -7200  10702  -7223  4855  11451
```

```
1210  7414  11219  11451  11453
```

 C:\Users\STRIX\source\repos\Lab8_ASD\Debug\Lab8_ASD.exe

```
Array size: 6
1353  5486  -129  -1663  3019  6446
5296  9686  -9392  -7088  -8075  11334
14086  -163  4869  -8518  -16000  -1621
-9946  -6545  12377  -8583  -13070  423
-7398  59  -7205  -13921  -7843  3635
-531  5312  1075  10305  -1453  -14055

1353  4869  9686
```

Висновки

Я дослідив алгоритм пошуку та сортування, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Побудував мат. модель, псевдокод та блок схему. Протестував алгоритм.