Реферативна частина проєкту з програмування

Хоменко Антон, комп’ютерна математика 1, 2 курс

Метод з використанням вказівників для виділення пам'яті для збереження матриць та векторів

SElement\_for\_vector: нова структура, об'єкт якої зберігає один індекс елемента (int i) та значення елемента (double value) вектора.

input\_element\_for\_vector: метод, що створює об'єкт типу Selement\_for\_vector, вводить з консолі індекс та значення елемента і повертає елемент.

show\_element\_from\_v: метод, що друкує об'єкт типу SElement\_for\_vector за шаблоном:

index: "(індекс елемента), value: (значення елемента)". Дана функція приймає як аргумент об'єкт типу SElement\_for\_vector.

SVector: нова структура, об'єкт якої містить розмір вектора (int n), кількість ненульових елементів вектора (int size\_all) та масив(SElement\_for\_vector \* vector) об'єктів типу SElement\_for\_vector(в цих об'єктах значення елемента(value) не є нульовим), що по суті і є розрідженим вектором.

key\_to\_value\_in\_vextor: метод, що приймає індекс елемента розрідженого вектора (int key), розріджений вектор (SVector vect), та повертає значення(value) елемента вектора з індексом key. Якщо vect не зберігає елемент з даним індексом, то функція повертає 0.

input\_vector: метод, що приймає об'єкт типу SVector, виділяє пам'ять, необхідну для одного елемента SElement\_for\_vector. Далі запускає цикл, в якому вводить елементи вектора та розширює пам'ять, доки не буде введений елемент із значенням 0 (SElement\_for\_vector.value = 0).

input\_vector\_from\_file: метод, що приймає об'єкт типу SVector та кількість елементів, що будуть введені з файлу(int size1), виділяє пам'ять, необхідну для цієї кількості елементів SElement\_for\_vector. Далі запускає цикл, в якому вводить елементи вектора з файлу, доки не буде введений елемент із значенням 0 (SElement\_for\_vector.value = 0) або не буде введена кількість елементів рівна size1.

show\_vector: метод, що приймає об'єкт типу SVector і друкує його елементи шляхом виклику (key\_to\_value\_in\_vector) значення елемента за кожним індексом

SElement: нова структура, об'єкт якої зберігає два індекси елемента (int i; int j) та значення елемента (double value) матриці.

input\_element: метод, що створює об'єкт типу SElement, вводить з консолі індекси та значення елемента і повертає елемент.

show\_element: метод, що друкує об'єкт типу SElement за шаблоном:

indexes: "(індекс елемента i, індекс елемента j), value: (значення елемента)". Дана функція приймає як аргумент об'єкт типу SElement.

SMatrix: нова структура, об'єкт якої містить розмір матриці (int n, int m), кількість ненульових елементів вектора (int size\_all) та масив(SElement \* matrix) об'єктів типу SElement(в цих об'єктах значення елемента(value) не є нульовим), що по суті і є розрідженою матрицею.

key\_to\_value: метод, що приймає масив з двох індексів елемента розрідженої матриці (int key[2]), розріджену матрицю (SMatrix matr), та повертає значення(value) елемента матриці з індексами key[1], key[2]. Якщо matr не зберігає елемент з даними індексами, то функція повертає 0.

input\_matrix: метод, що приймає об'єкт типу SMatrix, виділяє пам'ять, необхідну для одного елемента SElement. Далі запускає цикл, в якому вводить елементи матриці та розширює пам'ять, доки не буде введений елемент із значенням 0 (SElement\_for\_vector.value = 0).

input\_matrix\_from\_file: метод, що приймає об'єкт типу SMatrix та кількість елементів, що будуть введені з файлу(int size1), виділяє пам'ять, необхідну для цієї кількості елементів SElement. Далі запускає цикл, в якому вводить елементи матриці з файлу, доки не буде введений елемент із значенням 0 (SElement.value = 0) або не буде введена кількість елементів рівна size1.

show\_matrix: метод, що приймає об'єкт типу SVector і друкує його елементи шляхом виклику (key\_to\_value\_in\_vector) значення елемента за кожним індексом.

add\_element\_to\_matrix: функція, що приймає елемент (SElement a), матрицю (SMatrix \* matr), розширює пам'ять на кількість пам'яті, необхідну для одного елемента та додає елемент до матриці. За необхідності збільшує розміри матриці(matr.n, matr.m).

sum\_matrix: функція, що приймає дві розріджені матриці (SMatrix m1, SMatrix m2), створює матрицю sum\_mat, поелементнот обчислює елементи матриці суми m1 та m2 і записує їх в матрицю sum\_mat, якщо вони не рівні нулю. Повертає матрицю sum\_mat.

multiplication: функція, що приймає дві матриці (SMatrix m1, SMatrix m2), створює матрицю SMatrix mult\_mat, по черзі обчислює елементи матриці добутку m1 та m2 і записує їх матрицю mult\_mat, якщо їх значення не рівне нулю. Повертає матрицю добутку mult\_mat.

multiplication\_matrix\_and\_vector: функція, що приймає розріджену матрицю та розріджений вектор (SMatrix m1, SVector v1), створює матрицю mult\_mat, поелементнот обчислює матрицю добутку матриці m1 та вектора v1 та додає їх в матрицю mult\_mat, якщо вони не рівні нулю. Повертає матрицю mult\_mat.

det: рекурсивна функція, що приймає розріджену матрицю (SMatrix mat) та ще не обчислений визначник(double \* dett) та обчислює детермінант(визначник) матриці mat, шляхом обчислення суми детермінантів всіх можливих матриць, утворених шляхом викреслення одного рядка та одного стовбця з матриці mat. Повертає визначник (dett).

obernena\_matricya: функція, що приймає розріджену матрицю (SMatrix m1) та обчислює обернену їй матрицю (SMatrix ober\_mat) шляхом обрахунку матриці алгебраїчних доповнень, транспортування її та ділення кожного елемента на визначник матриці m1. Повертає матрицю ober\_mat.

swap\_two\_rows: функція, що приймає матрицю та два індекси (Matrix m1, int n1, int n2), створює матрицю swap\_mat та записує в неї елементи матриці m1, міняючи місцями рядки n1 та n2, просто змінюючи індекс i елементів цих рядків на відповідний. Повертає матрицю swap\_mat.

swap\_two\_collumns: функція, що приймає матрицю та два індекси (Matrix m1, int n1, int n2), створює матрицю swap\_mat та записує в неї елементи матриці m1, міняючи місцями стовбці n1 та n2, просто змінюючи індекс j елементів цих стовбців на відповідний. Повертає матрицю swap\_mat.

Метод з використанням контейнеру valarray для збереження матриць та векорів

Матриця зберігається як об’єкт vector<valarray<double> >. Відповідно, кожен її рядок це valarray;

input\_matrix: приймає розміри матриці (int n, int m), створює матрицю

(vector<valarray<double> > matr(n)), вводить з консолі елементи та повертає матрицю.

input\_matrix\_from\_file: робить теж саме, що й попередня функція, але елементи вводить з файлу “matrix.txt”.

show\_matrix: приймає матрицю, виводить елементи матриці як матрицю.

Вектор зберігається як об’єкт типу valarray<double>

input\_vector: приймає розміри вектора (int n, int m), створює вектор

(valarray<double> vect(n)), вводить з консолі елементи та повертає вектор.

input\_vector\_from\_file: робить теж саме, що й попередня функція, але елементи вводить з файлу “vector.txt”.

show\_vector: приймає вектор, виводить через пробіл його елементи

plus\_m: приймає дві матриці, створює матрицю matr3, обчислює суму двох матриць, записуючи результат в matr3, повертає матрицю суми matr3

determinante: приймає матрицю, перетворює її на верхню трикутну шляхом додавання та віднімання рядків (valarray), після зведення обчислює добуток діагональних елементів, повертає добуток, що також є визначником матриці.

multiply\_m1: приймає дві матриці, створює матрицю matr3, обчислює добуток двох матриць, записуючи результат в matr3, повертає матрицю добутку matr3

inverse\_matrix: приймає матрицю mat, створює матрицю inv\_mat, записує в неї транспоновані алгебраїчні доповнення, поділені на визначник матриці mat. Повертає матрицю inv, що є оберненою матрицею до mat

multiplies\_matrix\_and\_vector: приймає матрицю та вектор, обчислює їх добуток, повертає матрицю добутку

swap\_two\_rows: приймає матрицю та два індекси, міняє місцями рядки з відповідними індексами

swap\_two\_columns: приймає матрицю та два індекси, міняє місцями стовбці з відповідними індексами

Методи бібліотеки valarray

= призначає вміст

[ ] виклик елемента valarray з відповідним індексом

+, -, ==, != застосовуються одночасно до всіх елементів valarray

+=,-=, \*=, /=, %=, &=, |=, ^=, <<=, >>= застосовують складений оператор присвоєння до кожного елемента valarray

swap міняє місцями два valarray

size повертає розмір valarray

resize змінює розмір valarray

sum повертає суму всіх елементів valarray

min повертає мінімальний елемент valarray

max повертає максимальний елемент valarray

shift прибирає нульові елементи, зміщуючи інші на їх місце

cshift зміщує елементи по кругу

apply застосовує функцію до кожного елемента valarray

Результат порівняння швидкості роботи методів valarray та звичайного масива

Найскладніша функція в цих методах – це пошук оберненої матриці. Valarray знайшов обернену матрицю 100000 разів за 1.6 секунд. Інший метод знайшов обернена до тієї самої 100 разів за 1.8 секунд. Отже, метод, що використовує valarray робить це в 1125 разів швидшеі