**­Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра автоматизованих систем обробки інформації**

**і управління**

**Звіт**

з лабораторної роботи № 6 з дисципліни

«Алгоритми та структури даних 2. Структури даних»

„ **Проектування і аналіз алгоритмів внутрішнього сортування**”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-01 Галько Міла Вячеславівна*

**Перевірив**

(прізвище, ім'я, по батькові)

*Головченко М.М.*

Київ 2021

**Лабораторна робота № 6**

„ **Проектування і аналіз алгоритмів внутрішнього сортування**”

**Мета роботи** – вивчити основні методи аналізу обчислювальної складності алгоритмів внутрішнього сортування і оцінити поріг їх ефективності.

**Завдання**

Згідно варіанту №6, виконати аналіз алгоритму

сортування Шелла (*d* = послідовність Пратта) внутрішнього сортування на відповідність наступним властивостям:

* стійкість;
* «природність» поведінки (Adaptability);
* базуються на порівняннях;
* необхідність додаткової пам'яті (об'єму);
* необхідність в знаннях про структуру даних.

Записати алгоритм внутрішнього сортування за допомогою псевдокоду (чи іншого способу по вибору).

Провести аналіз часової складності в гіршому, кращому і середньому випадках та записати часову складність в асимптотичних оцінках.

Виконати програмну реалізацію алгоритму на будь-якій мові програмування з фіксацією часових характеристик оцінювання (кількість порівнянь, кількість перестановок, глибина рекурсивного поглиблення та інше в залежності від алгоритму).

Провести ряд випробувань алгоритму на масивах різної розмірності (10, 100, 1000, 5000, 10000, 20000, 50000 елементів) і різних наборів вхідних даних (впорядкований масив, зворотно упорядкований масив, масив випадкових чисел) і побудувати графіки залежності часових характеристик оцінювання від розмірності масиву, нанести на графік асимптотичну оцінку гіршого і кращого випадків для порівняння.

## Аналіз алгоритму на відповідність властивостям

Аналіз алгоритму сортування вставками на відповідність властивостям наведено в таблиці 3.1.

Таблиця 3.1 – Аналіз алгоритму на відповідність властивостям

|  |  |
| --- | --- |
| **Властивість** | **Сортування вставками** |
| Стійкість | + |
| «Природність» поведінки (Adaptability) |  |
| Базуються на порівняннях | + |
| Необхідність в додатковій пам'яті (об'єм) | - |
| Необхідність в знаннях про структури даних | + |

## Псевдокод алгоритму

**procedure** ShellSort()

getPrattNums(sequence.size())

**for** n = 0; n < Pratt.gaps.size(); n++ **do**:

**for** i = 0; i < Pratt.gaps[n]; i++ **do**:

**for** j = 0; j < sequence.size(); j += Pratt.gaps[n] **do**:

subVector.insert(subVector.begin(), sequence[j])

**end for**

SortSubVector(subVector)

int counter = 0;

**for** j = 0; j < sequence.size(); j += Pratt.gaps[n] **do**:

sequence[j] = subVector[counter]

counter++

**end for**

subVector.clear();

**end for**

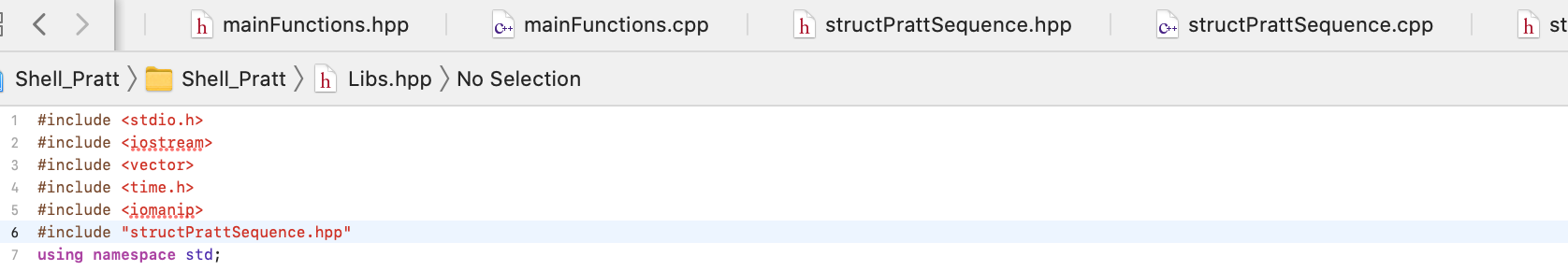
**end for**

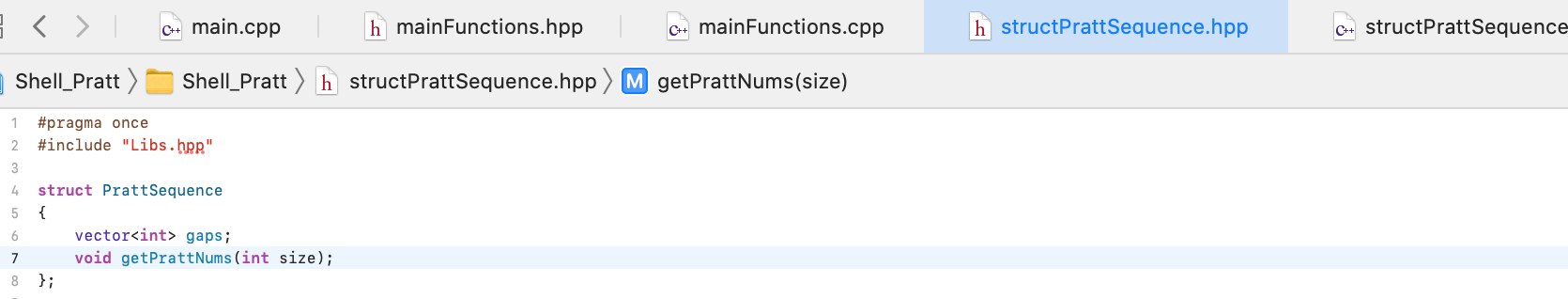
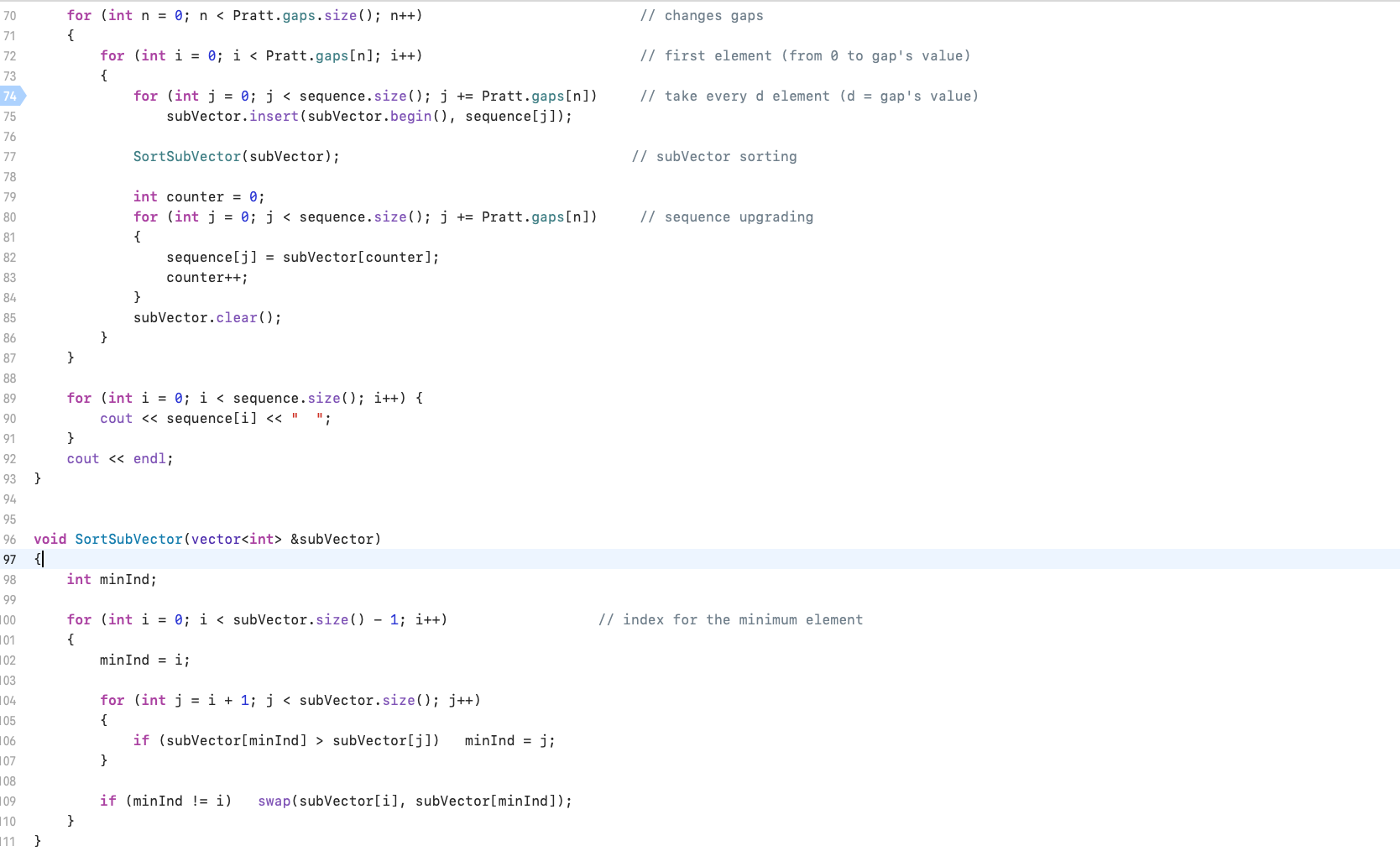
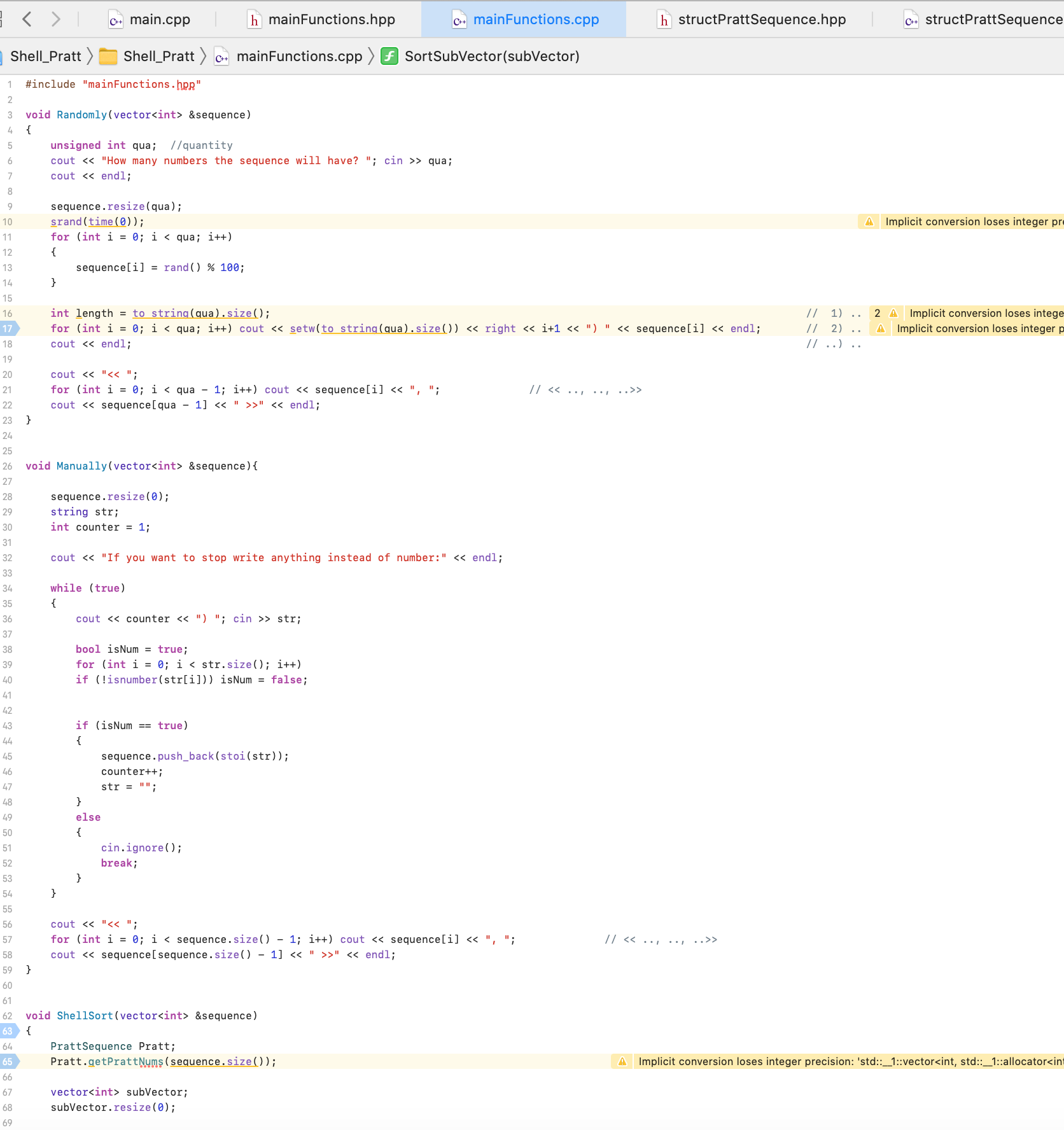
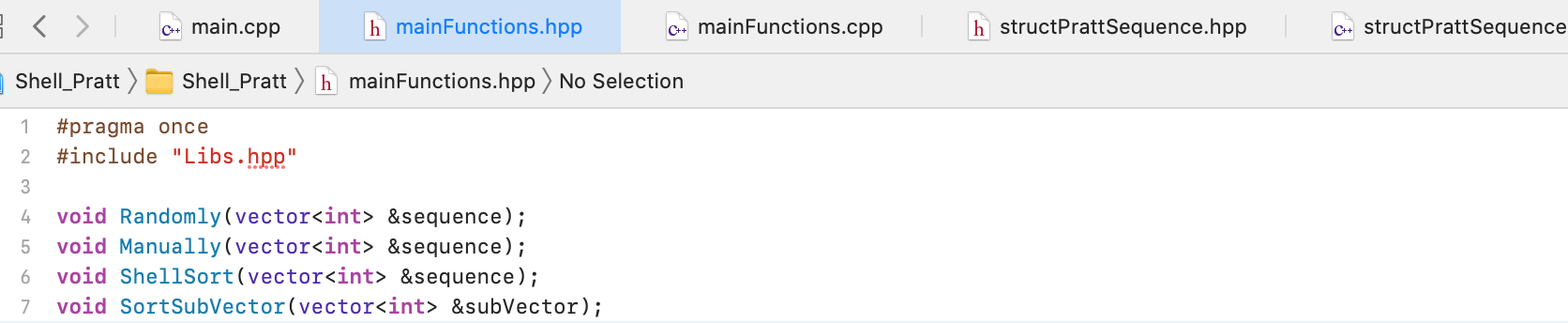
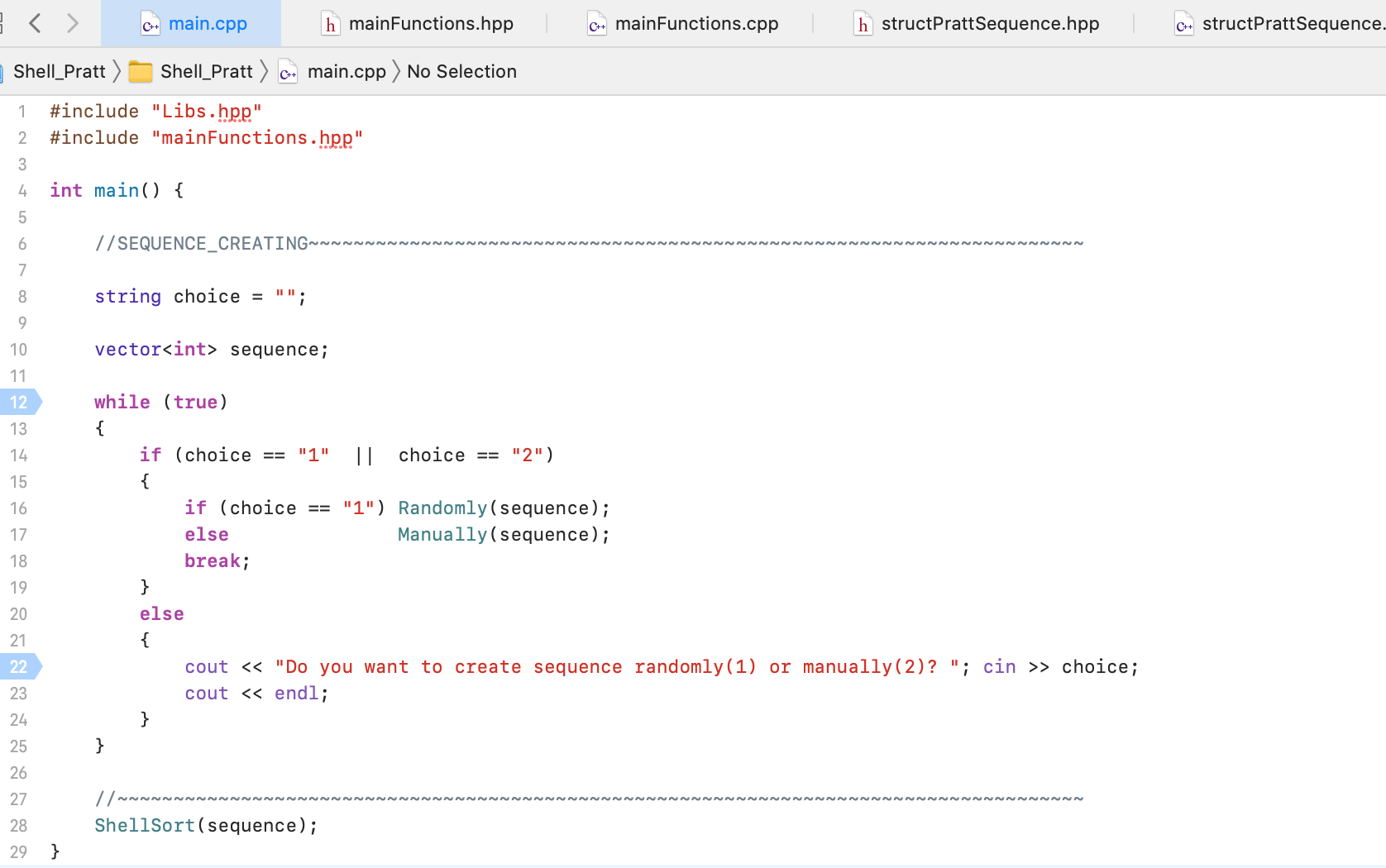
**end** procedure

## Аналіз часової складності

|  |  |  |
| --- | --- | --- |
| **for** n = 0; n < Pratt.gaps.size(); n++ | с1 | n |
| **for** i = 0; i < Pratt.gaps[n]; i++ | с2 |  |
| **for** j = 0; j < sequence.size(); j += Pratt.gaps[n] | с3 |  |
| subVector.insert(subVector.begin(), sequence[j]) | с4 | 1 |
| SortSubVector(subVector) | с5 | 1 |
| int counter = 0 | с6 | 1 |
| **for** j = 0; j < sequence.size(); j += Pratt.gaps[n] | с7 |  |
| sequence[j] = subVector[counter] | с8 |  |
| counter++ | с9 | 1 |
| subVector.clear(); | с10 | 1 |

## Програмна реалізація алгоритму





## 

## На 100 елементів:

**<< 56, 10, 31, 58, 25, 39, 50, 70, 69, 4, 54, 85, 67, 35, 44, 5, 18, 56, 7, 36, 30, 6, 14, 48, 18, 31, 8, 31, 40, 51, 18, 42, 27, 99, 30, 36, 82, 35, 36, 47, 94, 21, 93, 28, 8, 52, 12, 45, 97, 30, 84, 44, 10, 71, 27, 74, 61, 29, 42, 12, 58, 40, 2, 34, 21, 68, 48, 94, 30, 28, 20, 14, 53, 39, 9, 53, 95, 41, 45, 23, 29, 78, 49, 42, 12, 31, 25, 63, 51, 87, 88, 93, 9, 3, 28, 83, 65, 82, 72, 3 >>**

## **<< 2, 3, 3, 4, 5, 6, 7, 8, 8, 9, 9, 10, 10, 12, 12, 12, 14, 14, 18, 18, 18, 20, 21, 21, 23, 25, 25, 27, 27, 28, 28, 28, 29, 29, 30, 30, 30, 30, 31, 31, 31, 31, 34, 35, 35, 36, 36, 36, 39, 39, 40, 40, 41, 42, 42, 42, 44, 44, 45, 45, 47, 48, 48, 49, 50, 51, 51, 52, 53, 53, 54, 56, 56, 58, 58, 61, 63, 65, 67, 68, 69, 70, 71, 72, 74, 78, 82, 82, 83, 84, 85, 87, 88, 93, 93, 94, 94, 95, 97, 99 >>**

## Тестування алгоритму

### Часові характеристики оцінювання

В таблиці 3.2 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування вставками для масивів різної розмірності, коли масив містить упорядковану послідовність елементів.

Таблиця 3.2 – Характеристики оцінювання алгоритму сортування вставками для упорядкованої послідовності елементів у масиві

|  |  |  |
| --- | --- | --- |
| Розмірність масиву | Число порівнянь | Число перестановок |
| 10 |  |  |
| 100 |  |  |
| 1000 |  |  |
| 5000 |  |  |
| 10000 |  |  |
| 20000 |  |  |
| 50000 |  |  |

В таблиці 3.3 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування вставками для масивів різної розмірності, коли масиви містять зворотно упорядковану послідовність елементів.

Таблиця 3.3 – Характеристики оцінювання алгоритму сортування вставками для зворотно упорядкованої послідовності елементів у масиві.

|  |  |  |
| --- | --- | --- |
| Розмірність масиву | Число порівнянь | Число перестановок |
| 10 |  |  |
| 100 |  |  |
| 1000 |  |  |
| 5000 |  |  |
| 10000 |  |  |
| 20000 |  |  |
| 50000 |  |  |

У таблиці 3.4 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування вставками для масивів різної розмірності, масиви містять випадкову послідовність елементів.

Таблиця 3.4 – Характеристика оцінювання алгоритму сортування вставками для випадкової послідовності елементів у масиві.

|  |  |  |
| --- | --- | --- |
| Розмірність масиву | Число порівнянь | Число перестановок |
| 10 |  |  |
| 100 |  |  |
| 1000 |  |  |
| 5000 |  |  |
| 10000 |  |  |
| 20000 |  |  |
| 50000 |  |  |

### Графіки залежності часових характеристик оцінювання від розмірності масиву

На рисунку 3.3 показані графіки залежності часових характеристик оцінювання від розмірності масиву для випадків, коли масиви містять упорядковану послідовність елементів (зелений графік), коли масиви містять зворотно упорядковану послідовність елементів (червоний графік), коли масиви містять випадкову послідовність елементів (синій графік), також показані асимптотичні оцінки гіршого (фіолетовий графік) і кращого (жовтий графік) випадків для порівняння.

Рисунок 3.3 – Графіки залежності часових характеристик оцінювання

Висновок

При виконанні даної лабораторної роботи я вивчила основні методи аналізу обчислювальної складності алгоритмів внутрішнього сортування і зробила для алгоритму Шелла з послідовністю Пратта:

* аналіз на стійкість, «природну поведінку», порівняння, необхідність додаткової пам’яті та необхідність в знаннях про структуру даних;
* псевдокод алгоритму;
* аналіз часової складності в гіршому, кращому і середньому випадках;
* запис часової складності в асимптотичних оцінках;
* програму на С++ з можливістю формування послідовності, що сортується, випадковим та ручним чином;
* ряд випробувань алгоритму на масивах розмірності: 10, 100, 1000, 5000, 10000, 20000, 50000 елементів і різних наборів вхідних даних;
* графіки залежності часових характеристик оцінювання від розмірності масиву для гіршого і кращого випадків.