SWE3 HÜ 1: HeapSort & Komplexität

Erik Pitzer <erik.pitzer@fh-hagenberg.at>

Medizin- und Bio-Informatik – WS 2024/25

Name

Aufwand (in h)

Punkte

Aufgabe	Lösungsidee	Implementierung	Tests	Gesamtpunkte
1	40% / 20P	20% / 10P	40% / 20P	50
1	20% / 10P	30% / 15P	50% / 25P	50

Erstellen Sie für die folgenden Aufgaben eine gemeinsame Solution und die jeweilige Aufgabe als Projekt in einem Unterverzeichnis.

Aufgabe 1: Heap Sort Vervollständigen

Vervollständigen Sie die Implementierung und Dokumentation, fügen Sie mehr Testfälle hinzu und überlegen Sie Visualisierungen um den Aufbau des Heaps und die anschließende Sortierung besser zu veranschaulichen.

- Erklären Sie genau, wie der Algorithmus funktioniert, eventuell anhand von eigenen Beispielen
- Binden Sie relevante Quelltext-Snippets ein, anhand derer Sie den Algorithmus erklären
- Überlegen Sie neue Testfälle und protokollieren Sie die Ergebnisse

Aufgabe 2: Heap Sort Komplexität

Ersetzen Sie all Vergleichsoperationen zwischen Elementen des Vektors (`<`) durch eine geeignete Methode, z.B. `cmp` oder `less`, die sie selbst implementieren. Verfolgen Sie dann die Anzahl der Aufrufe. Dies kann am Einfachsten mit einer globalen Variablen gemacht werden.

Ersetzen Sie alle Aufrufe von `std::swap` durch eine eigene Implementierung, die ebenfalls die Anzahl der Aufrufe z.B. in einer globalen Variablen mit zählt.

Die beiden Funktionen könnten z.B. so aussehen:

```
// file: heap_sorter.h

extern int count_cmp;
extern int count_swap;

class heap_sorter {
        static bool less(const value_type& a, const value_type& b) {
                count_cmp++;
                return a < b;
        }

        static void swap(content_type &v, const index_type i, const index_type j) {
                count_swap++;
                std::swap(v[i], v[j]);
        }

};
```

```
// file: main.cpp

int count_cmp = 0;
int count_swap = 0;
```

Führen Sie dann mehrere Wiederholungen (idealerweise >=15) mit zufälligen Vektoren durch und vergleichen Sie die Komplexität von Vergleichs- und Änderungsoperationen mit der erwarteten Komplexität einer Heapsort-Implementierung in dem Sie verschieden große Arrays sortieren, z.B. 10, 100, 1 000, ... 100 000 000.

- Erklären Sie ihr Vorgehen!
- Dokumentieren Sie die Anpassungen!
- Bereiten Sie die Ergebnisse anschaulich auf (Tabelle, Charts, ...)!
- Ziehen Sie Schlussfolgerungen. Was kann man über die Laufzeitkomplexität sagen? Lassen sich die Vermutungen bestätigen?
- Können Sie auch aussagen zum Platzverbrauch machen?