2024 年 hoge 月 huga 日

データ構造とアルゴリズム

配列構造（ コンテナとソート ）1

Yugo Okamoto

目次

本資料は OECU プログラミングサークル (以下 OPC) の下級生メンバーに向けて, プログラミングの際に必要と思われるアルゴリズムとデータ構造を学ぶために作成した.

前期授業期間で錦氏 `@nkiib` やぶっとん氏 `@button501` が C/C++の基礎知識について解説してくれており, 前提として先述の講義を受けた者を本講義のターゲットとしている.

~~なお筆者の都合で他の言語による実装が登場するかもしれないが, やってることは C++とほとんど同じなので読み替えてもらって構わない.~~

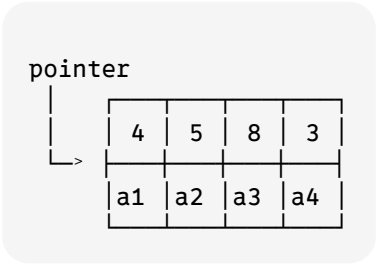
講義全体の内容は筆者の学科で「アルゴリズムとデータ構造」という授業が存在し, 基本的にはその講義の内容に沿いつつも一部で競技プログラミングの知識・問題を引用することもある.

1 配列構造

1.1 固定長配列

- メモリ上で指定した個数の変数をまとめて確保するデータ構造
  - 静的配列とも呼ばれる
- 互いの要素はメモリアドレス上で隣接している
- 配列の確保に成功すると**配列の先頭へのポインタが返される**

4 つの固定長配列を確保したときのイメージ図



- 要素の下側 a1 a4 はメモリアドレス
- ポインタは配列の先頭要素を表す

1.2 動的配列

- プログラム実行中でもサイズを変更できる配列
- 配列宣言時よりも余分にメモリを確保し, 足りなくなるとメモリを確保する処理を行う
  - 多くの場合は静的配列よりも実行速度が落ちる
- C 言語ではプログラマー自身が `malloc` 等を用いて確保しなければならない.

```c
int *arr;
arr = (int*)malloc(sizeof(int)* 11); // int 型配列を要素数 11 個で確保
```

- C++では `vector`, Rust では `vec!`, Python では `List` がそれに値する

```cpp
std::vector<int> arr(11, 0);   // int 型のベクタを要素数 11 個で確保. 同時に 0 で初期化
arr.push_back(3);              // 末尾に 3 を挿入
arr.pop_back();                // 末尾の要素を削除
int l = arr.size();            // 配列のサイズを取得
arr.insert(begin() + 2, 2);    // イテレータを指定して挿入
```

1.3 ソートアルゴリズム

- 配列の中身を昇順・降順に並べ替えるアルゴリズム
- 安定ソートは順位が同じである複数の要素がソート前後で入れ替わらない.

本講義で紹介するソートアルゴリズムは以下の通り

名称	平均計算時間	最悪計算時間	メモリ使用量	安定
バブルソート	n^2	n^2	1	○
クイックソート	$n \log n$	n^2	$\log n$	-
選択ソート	n^2	n^2	1	-
ヒープソート	$n \log n$	$n \log n$	1	-
挿入ソート	n^2	n^2	1	○
シェルソート	$n \log n$	$n^{\frac{3}{2}}$	1	-
マージソート	$n \log n$	$n \log n$	n	○

1.3.1 バブルソート

- 全ての要素に対して, 隣合う 2 つの要素を比較して順序が逆であれば入れ替える. これを要素数-1 回繰り返す.
- 平均・最悪計算時間共に$O(n^2)$なので効率は悪い
  - 実装が簡単かつ安定ソート

```cpp
#include <cstdlib>
#include <iostream>
#include <vector>

int main() {
  std::vector<int> arr = {6, 1, 3, 2, 8, 2, 4, 9};
  for (int i = 0; i < arr.size(); ++i) {
    for (int j = 1; j < arr.size() - i; ++j) {
      if (arr[j - 1] > arr[j]) {
        std::swap(arr[j], arr[j - 1]);
      }
    }
  }

  for (int i : arr) {
    std::cout << i << ", ";
  } // 1, 2, 2, 3, 4, 6, 8, 9,
  std::cout << std::endl;
  return EXIT_SUCCESS;
}
```

リスト 1: バブルソートの実装(C++)