sort

概要

sortはvectorなどの配列の要素の順番を変更するクラス

使用例

昇順

```
// 動的配列の宣言
vector<int> a{10,60,30,20,50};

// 昇順にsort
sort(a.begin(),a.end());
```

降順

```
// 動的配列の宣言
vector<int> a{10,60,30,20,50};

// 降順にsort
sort(a.begin(), a.end(), greater<int>());
```

特殊ソート

```
int main() {
    // 生徒のデータ（名前, 数学, 英語）
    using Student = tuple<string, int, int>;
    vector<Student> S = {{"Aoki", 13, 25},
                         {"Kobayashi", 64, 73},
                         {"Sato", 64, 62},
                         {"Suzuki", 87, 21},
                         {"Takahashi", 79, 60}};

    // 数学の点数が高い順を表す関数（get<1>() で数学の得点）
    auto cmp = [](Student x, Student y) -> bool {
        return get<1>(x) > get<1>(y);
    };

    // 数学の点数が高い順にソート
    sort(S.begin(), S.end(), cmp);

    // 出力
```

```
    for (Student s : S) {
        cout << get<0>(s) << ", " << get<1>(s) << ", " << get<2>(s) <<
endl;
    }
}
```

解説

- sort
  第１，第2引数にはイテレータを指定し、sortを実行する範囲を指定する。イテレータはコンテナ要素の位置を指定できる。第3引数は省略可能。省略した場合、昇順にソートされる。昇順以外の並び替えを行いたい場合は、greaterなどの関数オブジェクトを指定する。
- 関数オブジェクト
  通常の関数と同じ構文で関数呼び出しができるクラスである。最も単純なオブジェクトは次の通り。

  ```
  struct doubler {
    int operator ()(int x) const { return x * 2; }
  };
  ```

- greater()
  左辺と右辺の比較を返す

  ```
  int main()
  {
    std::cout << std::boolalpha << std::greater<int>()(3, 2) <<
  std::endl;
  }
  ```

  出力

  ```
  true
  ```

- begin, end
  コンテナクラスのデータ開始位置、終了位置のイテレータを返す。イテレータとは、実質ポインタのように扱える概念であり、コンテナを操作する際に使用する。コンテナは配列以外のデータ構造も存在し、ポインタを使用できないため、イテレータを使用する。
- STLコンテナ c++で標準用意されてる機能。テンプレートという機能を用いて様々なデータ型を格納できる機能である。様々なデータ型を使用できるので自由度が高い。各コンテナには共通の関数が用意されており短いコードで様々なことができる。コンテナには3種類ある。
  - シーケンスコンテナ
    - 要素の順番が保たれるコンテナ型。vevtor,list
  - 連想コンテナ
    - 要素の追加/削除のたびに自動整列するコンテナ型。set,map

    - 非連想コンテナ
        - 要素の追加/削除のたびに自動整列されないコンテナ型。unsorted_set,unsorted_map
    - stringクラス

参考

- 関数オブジェクト
- ソートの使い方
- イテレータとは
- STLコンテナとは