演習:list

- C++作業フォルダ内に**PracList**フォルダを作成
```
mkdir PracList
cd PracList
```

- **PracList**フォルダ内に main.cpp を作成する
```
copy nul main.cpp
```

- main.cpp をVisualStudioで開く

演習:list

- リスト同士の連結

int値を格納できるlistコンテナクラスの
インスタンス li1 と li2 を宣言し、それぞれ
初期値として次の数列を与える

```
li1 : 2, 3, 11, 17
li2 : 5, 7, 13, 19
```

このふたつのlistコンテナを用いて次の操作を行
うプログラムを作成しなさい

演習:list

- リスト同士の連結

① リストli1の後ろにリストli2の内容を連結して、
データ列が 2,3,11,17,5,7,13,19
になるようにする(※処理は後述)

② ①の処理の後、昇順に整列して
データ列が 2,3,5,7,11,13,17,19
になるようにする

演習:list

① リストli1の後ろにリストli2の内容を連結して、
li1の内容が 2,3,11,17,5,7,13,19
になるようにする

A) li1の末尾にli2の内容を先頭から順にひとつずつ
push_backすることで連結する
B) li2の先頭にli1の内容をデータ末尾から順に
ひとつずつpush_frontすることで連結する
C) insert, merge, spliceを用いて連結する

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //li1の末尾に追加して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    for (auto d : li2) {   //範囲forで各要素を順番に取り出す
        li1.push_back(d); //末尾に追加
    }
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
  { //li2の先頭に追加して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    //リバース(逆)イテレータを使ってli1の末尾から取り出す
    for (auto itr=li1.rbegin(); itr!=li1.rend(); itr++) {
        li2.push_front(*itr);
    }
    cout << “先頭に追加:”;
    for (auto d : li2) {
        cout << d << " ";
    }
    cout << endl;
  }
}
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.merge(li2);
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.insert(li1.end(),li2.begin(),li2.end());
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.splice(li1.end(),li2);
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

【実行結果】

```
末尾に追加:2 3 11 17 5 7 13 19
先頭に追加:2 3 11 17 5 7 13 19
mergeで連結:2 3 5 7 11 13 17 19
insertで連結:2 3 11 17 5 7 13 19
spliceで連結:2 3 11 17 5 7 13 19
```

演習:list

② ①の処理の後、昇順に整列して
li1の内容が 2,3,5,7,11,13,17,19
になるようにする

sort()を実行するだけでよい！

ただし、mergeしたものはすでに整列済みのため何もせずともよい

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //li1の末尾に追加して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    for (auto d : li2) {   //範囲forで各要素を順番に取り出す
        li1.push_back(d); //末尾に追加
    }
    li1.sort();
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
  { //li2の先頭に追加して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    //リバース（逆）イテレータを使ってli1の末尾から取り出す
    for (auto itr=li1.rbegin(); itr!=li1.rend(); itr++) {
        li2.push_front(*itr);
    }
    li2.sort();
    cout << “先頭に追加:”;
    for (auto d : li2) {
        cout << d << " ";
    }
    cout << endl;
  }
}
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.merge(li2);
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.insert(li1.end(),li2.begin(),li2.end());
    li1.sort();
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

PracList(main.cpp)

```
#include <list>
#include <iostream>
using namespace std;
int main() {
  { //mergeを使用して連結する方法
    list<int> li1{ 2,3,11,17 }, li2{ 5,7,13,19 };
    li1.splice(li1.end(),li2);
    li1.sort();
    cout << “末尾に追加:”;
    for (auto d : li1) {
        cout << d << " ";
    }
    cout << endl;
  }
```

演習:list

- メンバ関数を用いて2つのリストの連結が可能

- ただし、連結後のデータの並びは

  - merge:昇順に整列済み
  - insert:整列なし
  - splice:整列なし

  となる

演習:list

- 連結後のリストの状態
  - merge:li1に結合後、li2の内容は消える
  - splice:　　〃
  - insert:li2の内容はそのまま残る