# 令和6年度情報工学実験Ⅰ報告書

### 実験題目

プログラミング演習 4

### 指導教員

丸山教員,安細教員,周教員

### 実験日

● 令和6年10月02日(水)~令和6年10月16日(水)

# レポート

● 提出締切日: 令和 6 年 10 月 30 日 (水)

● 受理最終日: 令和6年11月20日(水)

● 提出日: 令和 6 年 \_\_\_\_\_ 月 \_\_\_\_ 日 (\_\_\_\_\_\_)

# 報告者

2年31番氏名橋本千聡

### 共同実験者

川和 李圭, 鈴木 隆生, 安田 れん

#### 1. 実験の目的

プログラムの共同開発演習を通して、議論などを伴うチームでのプログラム作成手法を理解する。

# 2. 実験の概要

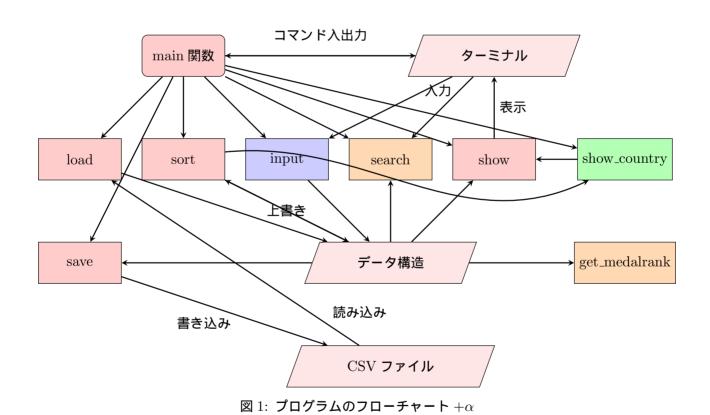
● 1・2 週目: 実行環境の確認及び C 言語サンプル実行確認 作成分担調整、分担一覧や全体構成の資料作成

◆ 3・4 週目: 各自の担当箇所を作成、単体動作確認
 ◆ 5・6 週目: 各自の作成の関数を統合して動作確認
 レポート報告内容のまとめ、レポート作成

### 3. 演習課題の報告

#### プログラム全体の概要

橋本 安田 川和 鈴木



#### 担当した機能の構成と説明

#### sort 関数

● 関数の説明: データ構造を並び替える

● 関数の入力: モード

● 関数の出力: なし(データ構造上書き)

● 関数の処理内容: merge sort を行い、mode によって国名金銀銅総メダル数のいずれかで並び替える

```
#include "main.h"
 1
2
      // 安定な比較関数
3
      int compare_by_mode(const country_data_type* a, const country_data_type* b,
4
          int mode) {
          if (mode == 0) {
5
              return strcmp(a->country, b->country); // 国名で昇順
6
          } else if (mode == 1) {
7
              return b->gold - a->gold; // 金メダル数で降順
          } else if (mode == 2) {
              return b->silver - a->silver; // 銀メダル数で降順
10
          } else if (mode == 3) {
              return b->bronze - a->bronze; // 銅メダル数で降順
12
          } else if (mode == 4) {
13
              return b->sum - a->sum; // 合計メダル数で降順
14
15
          return 0; // デフォルトは等しいと見なす
16
      }
17
18
19
      // 安定なマージ関数
      void merge(country_data_type arr[], int left, int mid, int right, int mode) {
20
21
          int n1 = mid - left + 1;
22
          int n2 = right - mid;
          country_data_type L[n1], R[n2];
23
          for (int i = 0; i < n1; i++)
24
              L[i] = arr[left + i];
25
          for (int j = 0; j < n2; j++)
26
              R[j] = arr[mid + 1 + j];
27
28
          int i = 0, j = 0, k = left;
29
30
          while (i < n1 \&\& j < n2) {
              if (compare_by_mode(&L[i], &R[j], mode) <= 0) {</pre>
32
                 arr[k] = L[i];
33
                 i++;
34
              } else {
35
                 arr[k] = R[j];
36
                 j++;
37
              }
38
              k++;
39
          }
40
41
          while (i < n1) {
42
              arr[k] = L[i];
43
              i++;
44
              k++;
45
          }
46
47
          while (j < n2) {
48
              arr[k] = R[j];
49
              j++;
50
```

```
k++;
          }
52
       }
53
54
       // マージソート関数
55
       void mergeSort(country_data_type arr[], int left, int right, int mode) {
56
           if (left < right) {</pre>
57
              int mid = left + (right - left) / 2;
58
              mergeSort(arr, left, mid, mode);
59
              mergeSort(arr, mid + 1, right, mode);
60
              merge(arr, left, mid, right, mode);
61
          }
62
       }
63
64
       // ソート関数
65
       void sort(int mode) {
66
          mergeSort(data, 0, data_size - 1, mode);
67
68
```

#### load 関数

- 関数の説明: CSV ファイルからデータを読み込む
- 関数の入力: ファイル名
- 関数の出力: なし(データ構造上書き)
- 関数の処理内容: 引数のファイル名から CSV ファイルを読み込み、データ構造に格納する

Listing 2: load 関数のコード

```
#include "main.h"
         void load(char* filename){
                      FILE *fp;
   4
                       char full_filename[256];
                       snprintf(full_filename, sizeof(full_filename), "./data/%s.csv", filename);
                      printd("load_file:_\%s\n", full_filename);
                      fp = fopen(full_filename, "r");
                      if(fp == NULL){
                                  printf("ファイルが開けません\n");
10
                                  return;
11
12
                      printd("load_data_start\n");
13
                       char buf [4] [100];
14
                       fscanf(fp, "%[^,],%[^,],%[^,],%s\n", buf[0], buf[1], buf[2], buf[3]);
15
                      printd("header: _\%s, 
16
                      data_size = 0;
17
                      while(fscanf(fp, "%[^,],%d,%d,%d\n", data[data_size].country, &data[data_size
18
                                   ].gold, &data[data_size].silver, &data[data_size].bronze) != EOF){
                                  printd("load\_data[\%d]: \_ \%s \_ \%d \_ \%d \_ \%d \_ \%d \_ n", \ data\_size, \ data[data\_size].
19
                                               country, data[data_size].gold, data[data_size].silver, data[data_size
                                               ].bronze, data[data_size].medal_rank);
                                  data_size++;
20
21
22
                      get_medalrank(data_size);
```

#### save 関数

● 関数の説明: データ構造を CSV ファイルに書き込む

● 関数の入力: ファイル名

● 関数の出力: なし

• 関数の処理内容: 引数のファイル名にデータ構造を書き込む

#### show 関数

● 関数の説明: データ構造を表示する

関数の入力: なし関数の出力: なし

● 関数の処理内容: printf のフォーマット機能を駆使してテーブル形式でデータ構造を表示する

#### main 関数

● 関数の説明: メイン関数

● 関数の入力: 引数 DEBUG の有無

● 関数の出力:標準出力

● 関数の処理内容: bash をベースとしたターミナルでのコマンド入力を受け付け、各関数を呼び出す

● その他: デバッグモードを有効にすると、各関数の詳細情報を表示する

担当した機能の単体テストの方法と結果

統合したプログラムの結合テストの方法と結果

4. 実験の感想