Answer on Question #74859 - Programming & Computer Science - C

Question 74859:

How to change the code so that it can sort:

```
struct SResult sample[] = { {"A1234", 10},  {"A1239", 5},
    {"A1394", 7}, {"A1434", 3}, {"A1454", 5}, {"A2884", 7}, {"A3235", 7},
    {"A4334", 9}, {"A4884", 2}, {"A6934", 5}, {"A7265", 7}, {"A9559", 3} };

void counting_sort(struct SResult scoreArr[], int N, int final[]) {
    int freq[11] = { 0 }, cfreq[11] = { 0 };
    int i, curScore;

    //1. Compute Frequency
    for (i = 0; i < N; i++)
        freq[ scoreArr[i].score ] ++;

    //2. Compute Cumulative Frequency
    cfreq[0] = freq[0];
    for (i = 1; i < 11; i++)
        cfreq[i] = cfreq[i-1] + freq[i];

    //3. Produce Final Position
    for (i = 0; i < N; i++) {
        curScore = scoreArr[i].score;
        final[ cfreq[ curScore ] - 1 ] = curScore;
        cfreq[curScore]--;
    }
}
```

Answer:

1) Assign indexes instead of scores: `final[ cfreq[ curScore ] - 1 ] = i .`

2) Use a macro to define the size: `int freq[RANGE + 1] = { 0 }, cfreq[RANGE + 1] = { 0 } .`

```
==========================================================================
#include "stdlib.h"
#include "stdio.h"

struct SResult {
   char ID[6];
   int score;
};

struct SResult sample[] = { {"A1234", 10},  {"A1239", 5},
   {"A1394", 7}, {"A1434", 3}, {"A1454", 5}, {"A2884", 7}, {"A3235", 7},
   {"A4334", 9}, {"A4884", 2}, {"A6934", 5}, {"A7265", 7}, {"A9559", 3} };

void counting_sort(struct SResult scoreArr[], int N, int final[]) {
   int freq[11] = { 0 }, cfreq[11] = { 0 };
   int i, curScore;

   //1. Compute Frequency
   for (i = 0; i < N; i++)
      freq[ scoreArr[i].score ] ++;

   //2. Compute Cumulative Frequency
   cfreq[0] = freq[0];
   for (i = 1; i < 11; i++)
      cfreq[i] = cfreq[i-1] + freq[i];

   //3. Produce Final Position
   for (i = 0; i < N; i++) {
      curScore = scoreArr[i].score;
      final[ cfreq[ curScore ] - 1 ] = curScore;
      cfreq[curScore]--;
```

```
    }
}

int main(void) {
    int *final = 0;
    unsigned i = 0;
    unsigned arrN = sizeof sample / sizeof sample[0];
    final = calloc(arrN, sizeof sample[0]);
    counting_sort(sample, arrN, final);
    for (i = 0; i < arrN; i++)
        printf("%d %s %d\n", final[i], sample[final[i]].ID,
                        sample[final[i]].score);
    free(final);
    return 0;
}
```

```
#include "stdlib.h"
#include "stdio.h"

struct SResult {
    char ID[6];
    int score;
};

struct SResult sample[] = { {"A1234", 10},  {"A1239", 5},
    {"A1394", 7}, {"A1434", 3}, {"A1454", 5}, {"A2884", 7}, {"A3235", 7},
    {"A4334", 9}, {"A4884", 2}, {"A6934", 5}, {"A7265", 7}, {"A9559", 3} };

#define MAX_SCORE 1000
```

```
// min = 1 or 0

void counting_sort(struct SResult scoreArr[], const int arrN, int final[]) {
   int freq[MAX_SCORE + 1] = {0}, cfreq[MAX_SCORE + 1] = {0};
   int i, curScore;

   //1. Compute Frequency
   for (i = 0; i < arrN; i++)
      freq[ scoreArr[i].score ] ++;

   //2. Compute Cumulative Frequency
   cfreq[0] = freq[0];
   for (i = 1; i < arrN; i++)
      cfreq[i] = cfreq[i - 1] + freq[i];

   //3. Produce Final Position
   for (i = 0; i < arrN; i++) {
      curScore = scoreArr[i].score;
      final[ cfreq[ curScore ] - 1 ] = i;
      cfreq[curScore]--;
   }
}

int main(void) {
   int *final = 0;
   unsigned i = 0;
   unsigned arrN = sizeof sample / sizeof sample[0];
   final = calloc(arrN, sizeof sample[0]);
   counting_sort(sample, arrN, final);
   for (i = 0; i < arrN; i++)
      printf("%d %s %d\n", final[i], sample[final[i]].ID,
```

```
                    sample[final[i]].score);
    free(final);
    return 0;
}
```

Answer provided by AssignmentExpert.com