ICS_Lab6_Report

Xiaoma

2022 年 12 月 27 日

实验目的

- 使用 C/C++ 复现 lab1-4
- 考虑到 lc-3 不支持 $*, /, \%, >>, <<$ 运算，所以只允许使用 $+, -, =, ++, --, ==, !=, <, >, <=, >=, \&, |,$
- 可以使用 **for , while , do while , if , continue , break , switch case**

实验原理

lab1

计算一个 16 位二进制数 a 的低 b 位的个数
依次将 a 的低 b 位和 1 进行与操作，来判断该位是否为 1。

Algorithm 1: HammingWeight of Lower B Bits

Input: a, b;

Output: weight;

```
weight = 0;
judge = 0
for i = 0; i < b; ++i do
    temp = a & judge;
    judge += judge;
    if temp != 0 then
        ++weight;
return weight;
```

lab2

计算

$$F(0) = F(1) = 1$$

$$F(N) = F(N-2)\%p + F(N-1)\%q \quad (2 \le N \le 1024)$$

$$p = 2^k \quad (2 \le k \le 10), 10 \le q \le 1024$$

采用滑动窗口的方法来求解该问题，**注意：**$F(N-1), F(N-2)$ **只有在求和的时候取余，而存储时不需要取余。**

Algorithm 2: myFib

Input: p, q, n;

Output: F(n);

```
num1 = 1
num2 = 1
for n = n - 1; n > 0; –n do
    temp1 = (p - 1) & num1;
    temp2 = num2;
    while temp2 > 0 do
        temp2 -= q;
    temp2 += q;
    F = temp1 + temp2;
    num1 = num2;
    num2 = F;
return F;
```

lab3

实现求字符串的最大重复子串。

通过一次遍历即可求解该问题，记录此时最大重复子串的长度，当出现新的重复子串时，更新最大值。

Algorithm 3: maxRepeating

Input: The string : str; Size of str : N;

Output: The max-size of repetitive substring : *max_len*;

```
right = str[0];
left = right;
max_len = 1;
temp = 1;
for i = 1; i < N; ++i do
    right = str[i];
    i += 1;
    if left == right then
        temp += 1;
    else
        if max_len < temp then
            max_len = temp;
        temp = 1;
    left = right;
if max_len < temp then
    max_len = temp;
return max_len;
```

lab4

对 16 个人的成绩的升序排列，并求出这 16 个人中获得评级 A，B 的数量。

为了提高代码效率，在排序的同时计算评级 A，B 的数量，从而需要获得降序的排名，而题目要求将成绩升序排列，故如果采用逆序的降序排列，既可以得到正序的升序排列，又可以在排序是计算评级 A，B 的数量。

Algorithm 4: maxRepeating

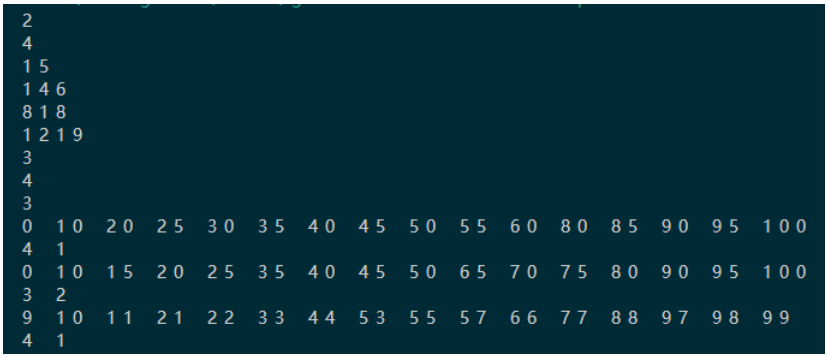
Input: The scores of students : scores[];

Output: The number of A and B: numa, numb;

```
numa = 0
numb = 0
for i = 15; i >= 0; –i do
    max_index = i;
    for j = i - 1; j >= 0; –j do
        if scores[max_index] < scores[j] then
            max_index = j;
    if i != max_index then
        swap(scores[max_index], scores[i]);
    if score[i] >= 85 && num_a < 4 then
        ++numa;
    else if score[i] >= 85 && num_a + num_b < 8 then
        ++numb;
    else if score[i] >= 75 && num_a + num_b < 8 then
        ++numb;
return numa, numb;
```

实验结果

执行给定的测试用例，结果为



实验思考

- 高级程序设计语言与 lc-3 汇编语言编程的区别
  - 高级语言可读性、可维护性较佳、代码简洁，lc-3 汇编语言的可读性较差、代码繁琐。
  - lc-3 汇编语言程序的占用空间小，执行速度快，执行效率高，高级语言占用的空间大，执行效率较低。
- 你认为 lc-3 汇编语言需要添加哪些指令
  - 可以添加取余相关的指令
- 对于使用的高级语言，是否需要从 lc-3 中学到什么
  - 通过使用 lc-3 进行面向寄存器的编程练习，提升了对程序执行时底层原理的理解，可以使我们在后续使用高级语言编程时，更好的提成程序执行效率。