

电子信息与通信学院

实 验 报 告

|  |  |
| --- | --- |
| 实验名称 | 课程综合练习 |
| 课程名称 | 计算机基础  与程序设计(C) |

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 何俊逸 | 学号 | U202411223 |

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 2024.12 | 地点 | 华中科技大学 |

|  |  |  |  |
| --- | --- | --- | --- |
| 成绩 |  | 教师 | 刘威 |

# 实验目的

完成大数计算系列代码。

# 实验环境

操作系统：Windows 11

编程工具：CodeBlocks 13.12

# 实验一

## 实验任务

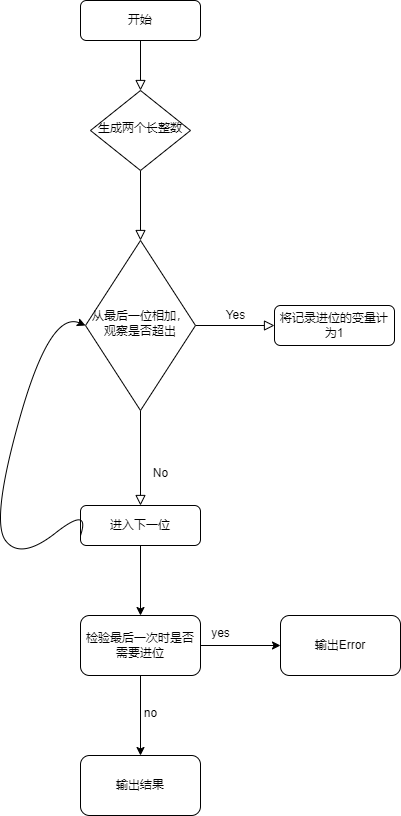
实验目的

* 掌握数值组这种数据结构的特点，并使用循环操作数值组的元素。
* 掌握通过数值组名类型函数参数的方法来实现基于引用的函数调用。

实验任务

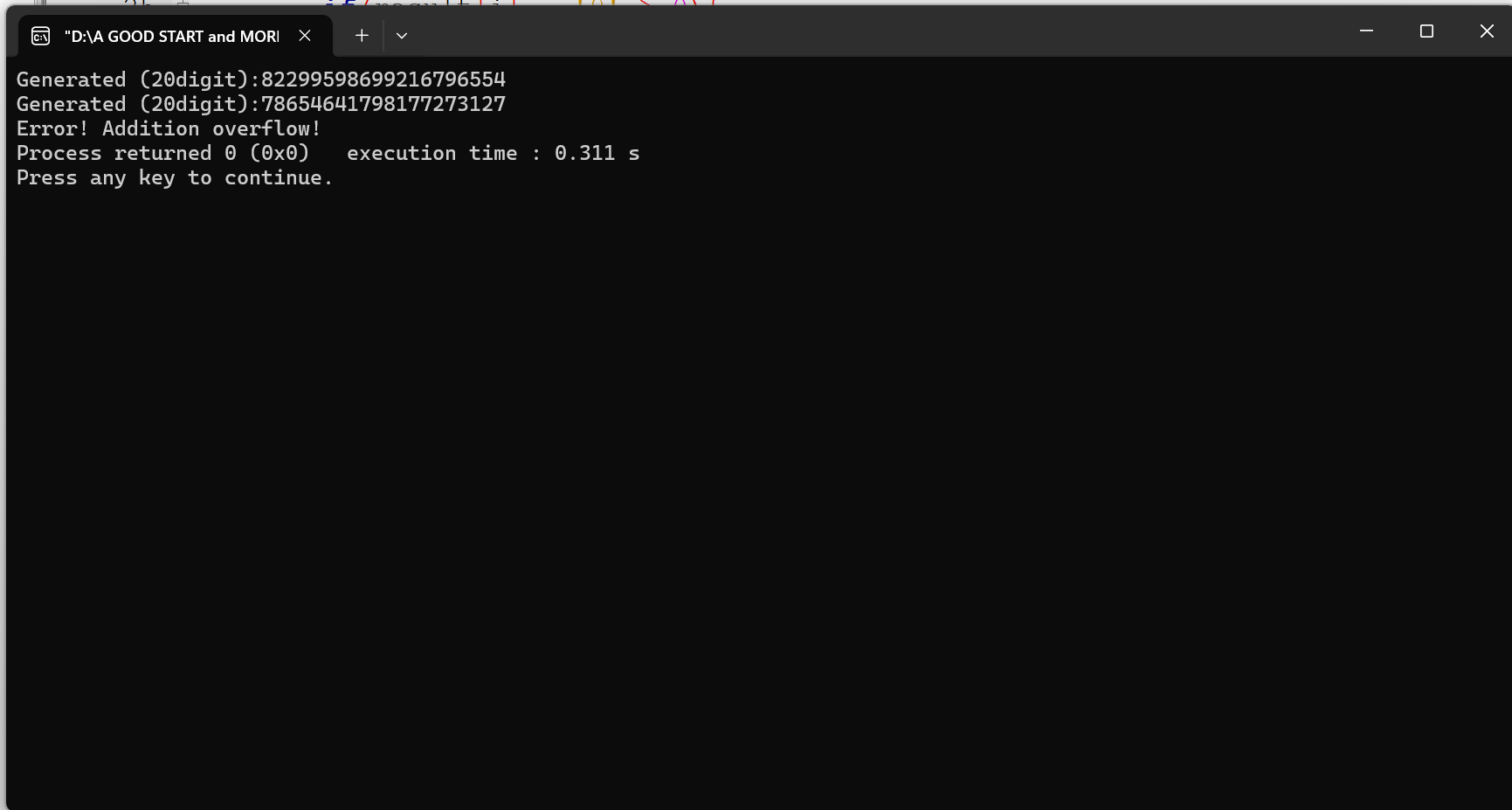
* 使用一维字符串数组记录长整数的每个数字位，在ASCII字符值转换中获取每个数字代表的实际数值。
* 随机生成两个长整数，但最高位不能为'0'
* 对相同长度的这两个长整数进行求和运算；如果存储位置不足导致无法完成进位，则需要输出错误信息。
* 打印出计算过程中的竖式结果。

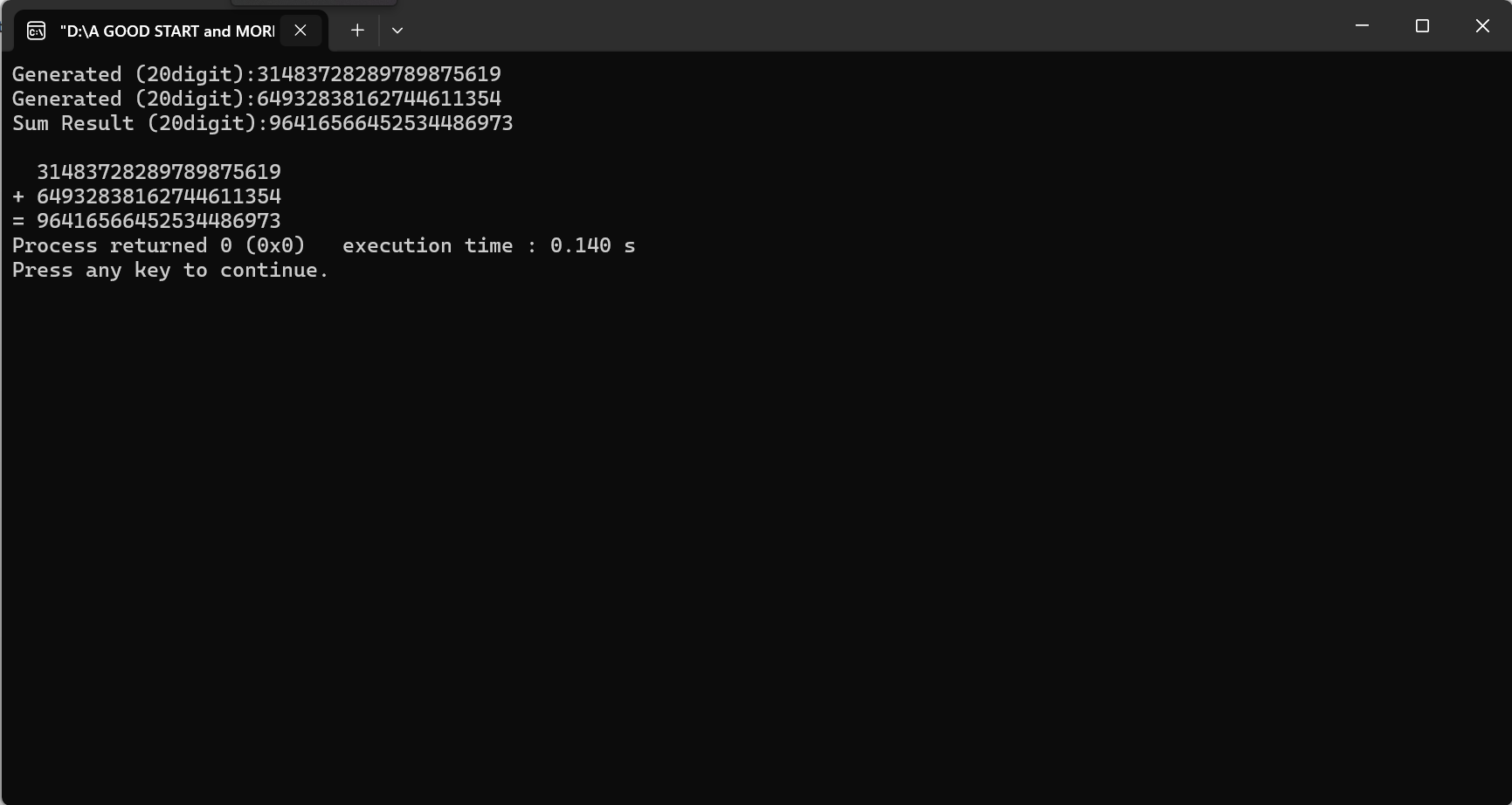
## 实验步骤



## 代码测试

### 测试点 输出的测试结果





实验达到测试点要求

## 实验结论

代码达到功能目标

## 实验总结

实验中我们实现了大数的求和，利用的是字符数组进行对每一位的储存，避免了直接使用变量时存在的溢出问题。

# 实验二

## 实验任务

**- 用一维字符数组记录长实数的每个数位,**

**- 通过ASCII字符值转换获取每个数位代表的数值,**

**- 用一个数位来存储小数点 '.'.**

**- 随机产生两个长实数,**

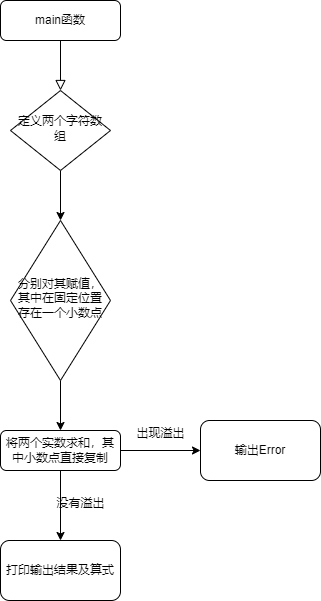
**- 约定其最高位不得为 '0'.**

**- 对两个相同长度的长实数进行求和,**

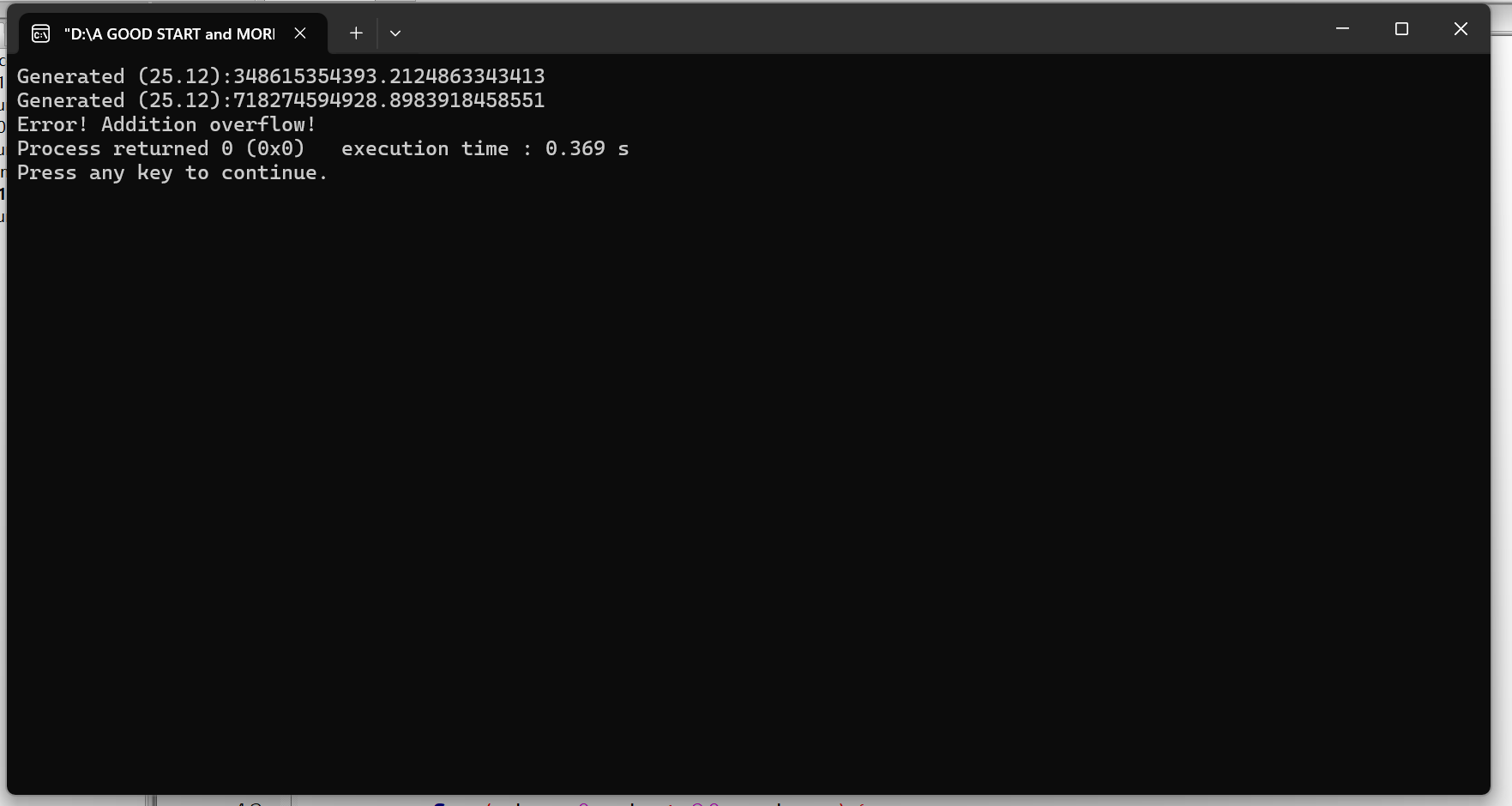
**- 有加法进位时显示报错信息.**

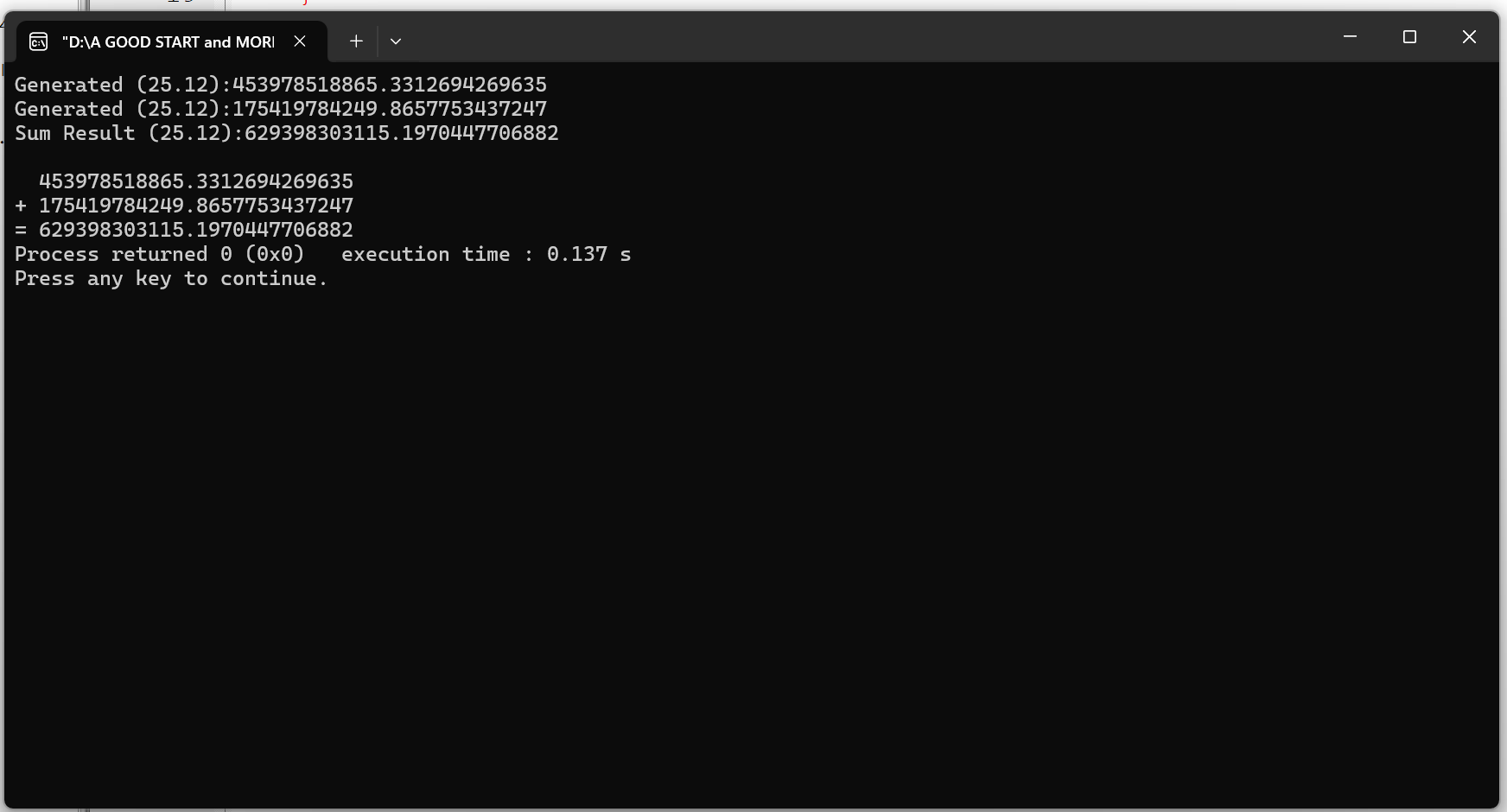
**- 打印两个长实数求和的竖式计算过程.**

## 实验步骤



## 4.3代码测试





4.3.1 测试点 输出的测试结果

代码呈现结果基本符合要求

4.4 实验结论

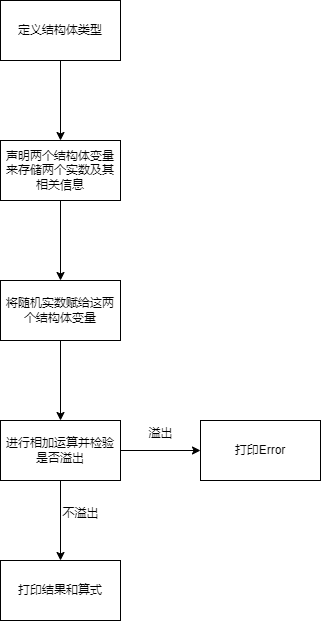
代码达到功能目标

4.5 实验总结

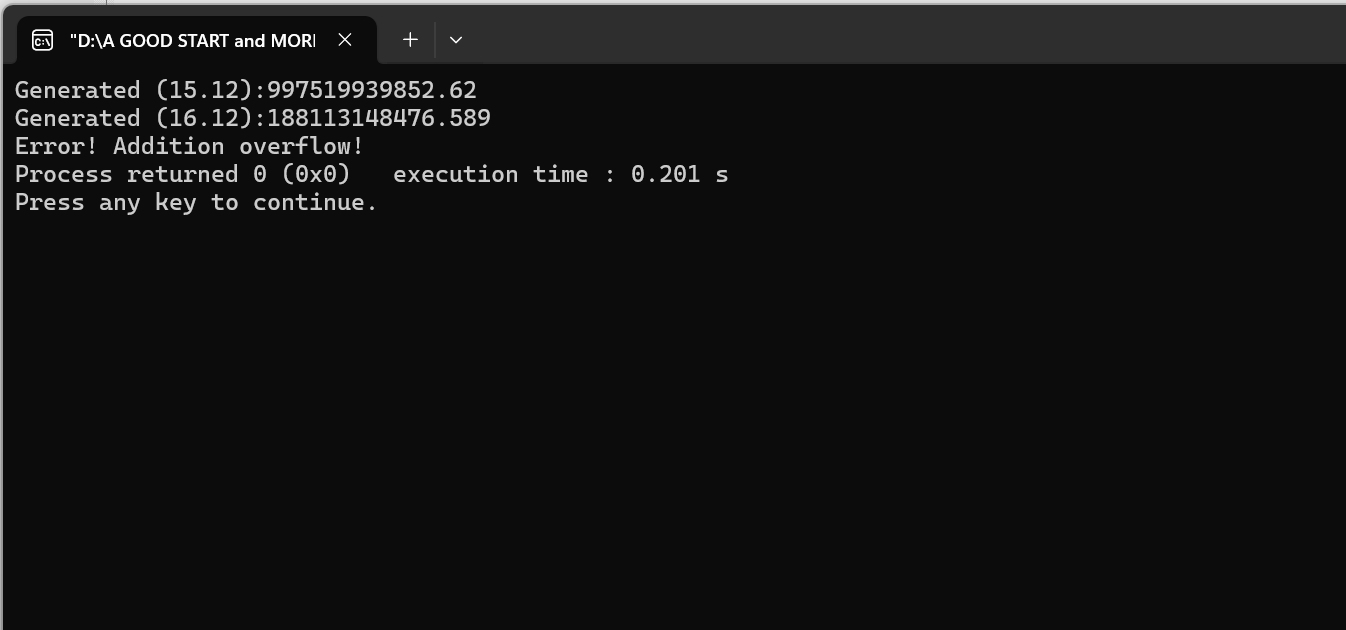
需要注意小数点的运算规则与数字不同，小数点应该直接复制，在相加过程中不能不加区分。

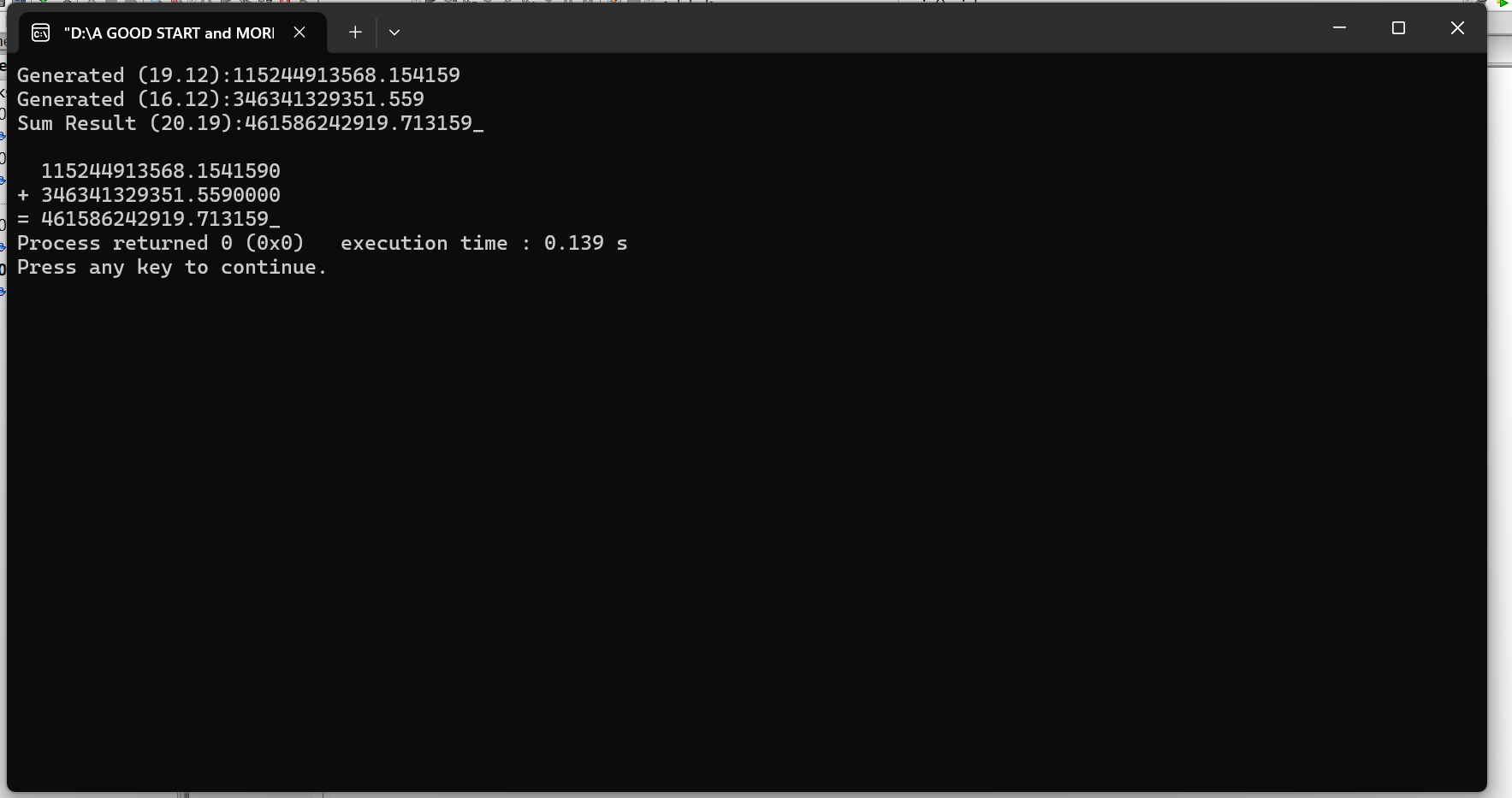
# 实验三

* 1. **实验任务**
* 结构体内，用一维字符数组记录长实数的每个数位，通过ASCII字符值转换获取每个数位，用一个数位来存储小数点‘.’，用一个整数记录长实数的长度。
* 随机产生两个长整数，约定其最高位不得为'0', 其长度可以不同, 但小数点位置相同.
* 对两个整数部分长度相同的长实数进行求和，并在加法进位时显示报错信息。
* 打印出两个长整数求和的竖式计算过程.
  1. **实验步骤**



* 1. **代码测试**
     1. 测试点 输出的测试结果





* 1. **实验结论**

代码达到功能目标

* 1. **实验总结**

因为结构体可以储存数字的长度，使得可以处理不同长度的实数相加，这个时候应当注意对多出来位数的处理，同时生成新的实数的时候可以利用结构体可以直接赋值的特性实现基于值的传递。

# 本课程学习总结

整个学习c语言的过程对我来说其实算得上是相当愉悦的，以至于我现在仍然有些意犹未尽（不是喊老师你 挂我重修的意思哈哈哈哈哈哈），或许如果让我对大一的所有课程中评价一门我最喜欢的课程，也许我会选择c语言，因为我的确很喜欢戴上耳机敲敲代码的感觉，就好像程序是一个完全由我来控制的世界，我去构建函数，去处理问题，所有的计算和操作都由所有的计算和操作都可以，都是由我来设想我来设计，就好像我在创造一个属于代码的世界一样，这样子的感觉让我非常非常喜欢这门课程，其实我也很有兴趣去深入的包括很多更加深入的东西，当然学习c语言也并不是完全一帆风顺的，他也并不是想象中那么轻松，尤其是刚开始学习的时候，相对来说比较陌生的环境比较愚蠢的错误，有时候犯一个愚蠢的错误，自己看不出来改代码改代码改了很长时间，没有发现就发现是很愚蠢的错误，有这也是的确挺气人的，挺懊恼的，还印象很让我很深刻的是一次写得代码，然后在我的电脑上是可以跑成功的，但是换了英特尔cpu的电脑就跑不成功了，这也让我有了一种观念，就是我必须要去把我的事情做得严谨，而不是他今天在我电脑上跑了能行，但因为他实际上是不严谨的，在其他电脑上跑不了，我也会在我以后的学习工作中去践行这一点，把所有的事情完善好，而不是做一个勉强能跑的东西，像是之后说的是什么屎山。然后其实我自己写代码，我也总结了一些小小的经验，特别是在代码调试的过程中，当然也是因为我总是写出一些不是特别好的代码，他容易出错，需要调试，第一个就是调试的时候可以使用printf这个函数去在屏幕中打印一些字符来提示你，特别是像复循环循环跑多少次这个程序运行到哪里不行了，这些都是可以去用printf这个函数在屏幕中打一些事情，让我们去意识到发生了什么，第2就是每次修改函数的时候修改程序的时候一定要注意保存程序，因为我有时候用CMD的命令行窗口去跑程序，但是不记得去做一个保存就会导致经常改了之后自己发现改不了而导致一些误判。然后就是一些我觉得自己还需要完善的地方，第一个我的代码书写是存在一定的规范性的，有的时候一些地方该家的空格，我是会存在一些遗漏问题，其实这个是相当不好的，总是在写完代码之后需要花大量的时间去进行一个格式的修改第二个就是我有的时候会缺少一些写注释的习惯，容易让自己的大脑去跑飞了，但是不给未来自己也不给其他人读自己的代码的机会，这其实是相当不好的，这一点是我需要记住，我就是在代码层面上表达好自己的想注释层面去实现一个很好的阐述和表达，这也就是老师之前曾经讲过的注释也是代码的一部分，也是需要认认真真好写好的，我会努力改正这两点问题

# 附录

* 1. 实验一
* #include <stdio.h>
* #include <stdlib.h>
* #include <time.h>
* **int** main()
* {
* **char** a[20] = {0},b[20] = {0};
* srand(time(NULL));
* **int** i;
* printf("Generated (20digit):");
* **for**( i = 0; i < 20 ; i++ ){
* a[i] = rand()%9+'1';
* printf("%c",a[i]);
* }
* printf("\n");
* printf("Generated (20digit):");
* **for**( i = 0; i < 20 ; i++ ){
* b[i] = rand()%9+'1';
* printf("%c",b[i]);
* }
* printf("\n");
* **int** extra = 0;
* **char** result[20];
* **for**(i = 19; i >= 0; i-- ){
* result[i] = a[i] + b[i] -'0'+extra;
* **if**(result[i] - '9' > 0){
* result[i] -= 10;
* extra = 1;
* }
* **else**{
* extra = 0;
* }
* }
* **if**(extra == 1){
* printf("Error! Addition overflow!");
* }
* **else**{
* printf("Sum Result (20digit):");
* **for**( i = 0; i < 20 ; i++ ){
* printf("%c",result[i]);
* }
* printf("\n\n");
* printf("  ");
* **for**( i = 0; i < 20 ; i++ ){
* printf("%c",a[i]);
* }
* printf("\n+ ");
* **for**( i = 0; i < 20 ; i++ ){
* printf("%c",b[i]);
* }
* printf("\n= ");
* **for**( i = 0; i < 20 ; i++ ){
* printf("%c",result[i]);
* }
* }
* **return** 0;
* }

实验二：

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <time.h>
4. #define REAL\_WIDTH 25
5. #define POINT\_POSITION 12
6. **int** main()
7. {
8. **char** a[26] = {0},b[26] = {0};
9. srand(time(NULL));
10. **int** i;
11. printf("Generated (25.12):");
12. **for**( i = 0; i < 26 ; i++ ){
13. **if**(i == 12){
14. a[i] = '.';
15. printf("%c",a[i]);
16. **continue**;
17. }
18. a[i] = rand()%9+'1';
19. printf("%c",a[i]);
20. }
21. printf("\n");
22. printf("Generated (25.12):");
23. **for**( i = 0; i < 26 ; i++ ){
24. **if**(i == 12){
25. b[i] = '.';
26. printf("%c",a[i]);
27. **continue**;
28. }
29. b[i] = rand()%9+'1';
30. printf("%c",b[i]);
31. }
32. printf("\n");
33. **int** extra = 0;
34. **char** result[26];
35. **for**(i = 25; i >= 0; i-- ){
36. **if**(i == 12){
37. result[i] = '.';
38. **continue**;
39. }
40. result[i] = a[i] + b[i] -'0'+extra;
41. **if**(result[i] - '9' > 0){
42. result[i] -= 10;
43. extra = 1;
44. }
45. **else**{
46. extra = 0;
47. }
48. }
49. **if**(extra == 1){
50. printf("Error! Addition overflow!");
51. }
52. **else**{
53. printf("Sum Result (25.12):");
54. **for**( i = 0; i < 26 ; i++ ){
55. printf("%c",result[i]);
56. }
57. printf("\n\n");
58. printf("  ");
59. **for**( i = 0; i < 26 ; i++ ){
60. printf("%c",a[i]);
61. }
63. printf("\n+ ");
64. **for**( i = 0; i < 26 ; i++ ){
65. printf("%c",b[i]);
66. }
67. printf("\n= ");
68. **for**( i = 0; i < 26 ; i++ ){
69. printf("%c",result[i]);
70. }
71. }
72. **return** 0;
73. }

实验三：

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <time.h>
4. #define REAL\_WIDTH 20
5. #define POINT\_POSITION 12
6. **typedef** **struct** Real{
7. **char** big[20];
8. **int** length;
9. }real;
11. **int** main()
12. {
13. real a,b;
14. srand(time(NULL));
15. **int** i;
16. a.length = rand()%9+12;
17. b.length = rand()%9+12;
18. printf("Generated (%d.12):",a.length);
19. **for**( i = 0; i < a.length ; i++ ){
20. **if**(i == 12){
21. a.big[i] = '.';
22. printf("%c",a.big[i]);
23. **continue**;
24. }
25. a.big[i] = rand()%9+'1';
26. printf("%c",a.big[i]);
27. }
28. **for**(i = a.length;i<20;i++){
29. a.big[i]='0';
31. }
32. printf("\n");
33. printf("Generated (%d.12):",b.length);
34. **for**( i = 0; i < b.length ; i++ ){
35. **if**(i == 12){
36. b.big[i] = '.';
37. printf("%c",b.big[i]);
38. **continue**;
39. }
40. b.big[i] = rand()%9+'1';
41. printf("%c",b.big[i]);
42. }
43. **for**(i = b.length;i<20;i++){
44. b.big[i]='0';
46. }
47. printf("\n");
48. **int** extra = 0;
49. real result;
50. **for**(i = 19; i >= 0; i-- ){
51. **if**(i == 12){
52. result.big[i] = '.';
53. **continue**;
54. }
55. **if**(i >= (a.length>b.length?a.length:b.length)){
56. result.big[i] = '\_';
57. **continue**;
58. }
59. result.big[i] = a.big[i] + b.big[i] -'0'+extra;
60. **if**(result.big[i] - '9' > 0){
61. result.big[i] -= 10;
62. extra = 1;
63. }
64. **else**{
65. extra = 0;
66. }
67. }
68. **if**(extra == 1){
69. printf("Error! Addition overflow!");
70. }
71. **else**{
72. printf("Sum Result (20.%d):",a.length>b.length?a.length:b.length);
73. **for**( i = 0; i < 20 ; i++ ){
74. printf("%c",result.big[i]);
75. }
76. printf("\n\n");
77. printf("  ");
78. **for**( i = 0; i < 20 ; i++ ){
79. printf("%c",a.big[i]);
80. }
82. printf("\n+ ");
83. **for**( i = 0; i < 20 ; i++ ){
84. printf("%c",b.big[i]);
85. }
86. printf("\n= ");
87. **for**( i = 0; i < 20 ; i++ ){
88. printf("%c",result.big[i]);
89. }
90. }
91. **return** 0;
92. }