DOI: 10.23977/acss.2023.070504
ISSN 2371-8838 Vol. 7 Num. 5

Analysis of Computer C Language Program Technology—Taking Vertical Division as an Example

Jiaying Zhu

Qingdao No.58 High School Shandong Province, Qingdao, Shandong, China

Keywords: Computers; C language programming technology; Vertical division

Abstract: The rapid development of computer technology has made the programming functions of C language more perfect, directly enhancing expression ability, and having strong flexibility and functionality in use. There are differences between computer C language programming technology and the commonly mentioned preloading, and it is necessary to enable the application to understand the usage methods and grammar knowledge of C language, in order to successfully complete code editing and achieve a certain software function. Based on this, the article first introduces the process of editing and running computer C language programs, proposes common errors in computer C language program editing, and then uses vertical division as an example to analyze the design and implementation of computer C language program technology, hoping to provide some reference for everyone.

In the specific application of computer C language, it not only has all the characteristics of High-level programming language, but also has the advantages of good indirection. It is a systematic description language, and is also a common language mode in the programming stage. Computer talents are bound to master this technology. The vertical division in computer C language can present a starting calculation process, prevent the occurrence of calculation errors, and is widely used in work, learning, and daily life. In this regard, the establishment of a vertical division simulator can be achieved with the help of computer C language programming technology, understanding the array of C language, and using function knowledge to carry out experimental design, reflecting new engineering thinking. It is greatly helpful for using C language and learning C language well, and will cultivate more computer talents.

1. Editing and Running Computer C Language Program Procedures

Y For the edited computer C language program code, simply input it into the computer and compile it into the target program file. obj, executable program file. exe. The program is generated by running the C language program file. In the stage of using C language, editing and design work is very complex, and it is necessary to manually operate on the computer to prevent errors and ensure stable operation of C language programs. During the program runtime, connection processing should be carried out, not only by directly compiling, but also by being able to run directly. After the operation is completed, determine the storage location of the source file and close the program [1].

2. Common Mistakes in Computer C Language Programming

2.1 Variable editing error

The first type is to forget to define variables.

For example:

```
Void main 0{
X=3;
Y=5;
Printf ("% d  n", x+y);
}
```

In the application stage of computer C language, during program design and editing, it is necessary to define each variable used in the program. However, during program design, the definitions of x and y were not completed. In the correct editing process, the following will be added at the beginning of the function: int x, y; Ensure the integrity of C language programming [2].

The second type, when designing computer C language programs, there are differences in the data format of C language.

For example, defining A as an integer and B as a real type, the content is as follows:

```
A=3
B-4.5/* The assignment operation of A and B in the program*/
Printf ("% f% d  n". A, B);
```

Regarding this, during the C language compilation stage of the computer, the output result obtained is 0.00000016402, so it is necessary to convert 4.5 to 4.

The third type, the failure to use the address character "&" when outputting computer C language, is a very common programming error, such as:

Scanf ("% d% d", a, b); To be written as scanf ("% d% d",&a,&b);

2.2 C language array editing error

When programming in C language, parentheses are used to introduce some array elements:

```
Voidmain ({inti, a [1o];
For (i=0; i<10; i++)
Scanf ('% d'.&a (i));}
```

The method of referencing multidimensional and two-dimensional arrays is incorrect and imprecise in definition, as reflected in the following:

```
Void main()
{inta [5,4];
}
```

The above array error needs to be corrected as a {5} {4};

2.3 Errors in C Language Pointer Editing

This type of error can be reflected in two aspects, namely the confusion between C language character arrays and pointers, and the failure to assign values when introducing C language pointer variables. The complete content is shown in Table 1.

Table 1: Errors in C Language Pointer Editing

types of	content

1. Character arrays and indicators in C language are confused.	Void main()
	{
	Char str{4};
	Str=”Compute and c”;
	}
	In C language, str is the array name, representing the first address of the array. During runtime, str failed to assign a value. For this, it needs to be adjusted to "char * str" as the character pointer, and the program can run normally at this time.

2. Unassigned pointer variables in C language applications	Viod main()
	{char *p;
	Scanf(“%s,p”);
	}
	When designing and editing computer C language programs, if there is no direct assignment of values, pointers cannot be directly used. If pointer variables are directly used, C language editing will generate error messages, which should be adjusted to:
	Char * p, c ("% s, P");
	P=c;
	Scanf ('% s, p')
	After running, the C language program is correct.

3. Taking Vertical Division as an Example—The Path of Computer C Language Programming and Implementation

When using vertical division C language programming technology, the main operating rules are as follows: first, leave a blank line to print the quotient, then print a separator line between the quotient and dividend in the second line, and input the divisor, divisor, and dividend in the third line [3]. Next, output the difference, product, and new dividend of the divisor and quotient. Repeat the above steps to obtain the final result. For example, "58327 ÷19=".

3.1 Function diagram

By studying the calculation rules and process of vertical division, the construction of the function diagram is completed. Generally, two random numbers will appear, and a random integer within a certain range will be generated through a random function. At this time, the large number is the dividend, and the small number is the divisor. In order to generate different numbers for each calculation program, a random number generator seed, such as srand (unsigned int) time (NULL), needs to be designed to form numbers through random functions.

3.1.1 Storage and separation of integers

This section is the basic prerequisite for the following operations. The separation of integers, which involves dividing multiple integers into one number after another, involves modular operations. For the separated numbers, they can be stored through an array. Once the actual number of elements in the array is determined, the number of digits is the integer.

3.1.2 Initial state of output vertical division

When using C language programs, the main purpose of outputting a blank line is to reserve the position for quotient output, and then output the horizontal line separating the dividend and quotient. Next, we need to output two random integers representing the division sign between the divisor and the dividend. The earliest output met the style and rules of vertical division, which also prepared for subsequent output and calculations.

3.1.3 Output and Calculation

The above modules cover many functions, but there are two main ones. The functions will be interdependent, and the use of functions is also a key point in the application of C language programs. Strict calculations and deductions should be carried out to ensure that there is no misalignment.

In the functional section, not only are the functions to be achieved in the experiment presented, but also the execution order of each function is listed, one for the calculation function and the other for the output function. When calculating while outputting, the quotient can be immediately obtained, and then the product can be calculated to complete bitwise output. The new dividend can be obtained, and the final calculation and output can be repeated.

3.2 Detailed coding and design

3.2.1 Storage and data involved

In computer C language programming, for the direct contact of the dividend and the divisor, several constants, arrays, and simple variables need to be used in the operation stage. The specific content is shown in Table 2.

Table 2: Complete contents of constants, simple variables, and arrays

first	Maxlen indicates the maximum number of digits in the quotient, which is the numerical value of 10; The length of the space string is represented by Blan Klen, with a value of 5 bits.
second	A variable that stores two random integers, namely num1 and num2.
third	When num1 and num2 are stored separately, the digits of each digit need to be saved through numC1 [MaxLen] and numC2 [MaxLen]. Specifically, it means storing the digits in one digit with subscript 0 and the digits in ten digits with subscript 1.
fourth	The integer of the storage provider, i.e. resuIt [MaxLen].
fifth	Store the current remainder array and indicate that the remainder is invalid when the quotient is 0.

To ensure the accuracy of division, it is necessary to design limiting factors in advance and scientifically control the range of random number values to make the calculation results legal.

3.2.2 Generate two random integers on

By using random functions and random number seed generators, two random integers within a specified range are generated. After exchange processing, "num1 >=num2" is obtained, which means that the larger number is treated as the dividend and the smaller number is treated as the divisor to avoid division or the dividend being 0, so that the operation rules can be met.

3.2.3 Finding Remainder and Quotient

To solve for each remainder and quotient, the specific algorithm is as follows: the first quotient should form a new dividend from the first few digits of the dividend, so that the new dividend is not lower than the divisor, and then record the position of the quotient at this time. After the quotient is determined, start from 0 and add 1 each time, until the quotient is multiplied by 1 and the value of the divisor exceeds the dividend. Using a new dividend that does not exceed the divisor, determine the quotient position, and then store the current remainder and quotient. The integer 'num' can be used as a positional identifier, and the awareness of different value identifiers also varies. For example, -1 represents the initial state, and -2 represents the end of calculation.

3.2.4 Bitwise output

In this section, there are three specific output functions that also differ in functionality: ① show1() function. The Show1() function is the initial form of printing out a vertical division by 2, which outputs the dividend, divisor, and division sign based on the style of the vertical division S_ The printf() function. Add s_ The printf function is customized and has two functions: printing each digit of the product and remainder. In order to align the positions of each number output, a custom coordinate positioning can be used to define the delay function, which will reflect the overall process of surgical calculation.

3.2.5 Show 2() function

Function refers to the quotient, difference, product, and new dividend calculated at each step after output.

To solve the vertical division problem through a computer C language program, it is necessary to build a simulator, master key operating points, and set the program in accordance with computational requirements. In the specific editing and design, not only the position of the Decimal separator should be considered, but also the accuracy of quotient should be examined. Therefore, the implementation is more complex, which is also the content of subsequent research.

In summary, when applying computer C language programming technology, it is necessary to understand the functions of computer control systems, understand the relevant issues and principles of C language design, and thus promote programming activities. This article takes vertical division as an example and uses C language programs to establish a simulator, achieve key operations, and implement the application of various algorithms. Through in-depth analysis of computer C language programming technology, people's skill level can be improved, the application value of C language can be realized, the homework pressure of computer C language programming personnel can be reduced, and the efficiency of computer C language application can be improved.

References

[1] Zhang Xiu. Innovative research on teaching mode of "C language programming" course based on Big data technology [J]. Wireless Internet Technology, 2022, 19 (1): 145-146

[2] Wang Qirui, Zhang Qingguo. Research on Hybrid Programming Technology of Assembly Language and C Language [J]. Computer Programming Skills and Maintenance, 2021 (9): 44-45, 53

[3] Ke Liangliang. Teaching Reform and Practice of the Course "C Language Programming" in Higher Vocational Education under the New Situation—Taking Tongling Vocational and Technical College as an Example [J]. Journal of Yan'an Vocational and Technical College, 2019, 33 (3): 68-72