Application of Visual Programming in Program Design Course

Xiao-jun BAI and Bai-lin LIU

School of Computer Science and Engineering, Xi’an Technological University, Xi’an, China

Keywords: Visual Programming, Computer programming, Computational thinking.

Abstract. According to the problems in teaching of computer programming course, brought visual programming tools to assist teaching, introduced the advantages of visual programming, and made use of it in algorithm design, coding and modular design. Practice shows that by introducing of visual programming, it helps the students to improve their interesting, and develop the ability of logic thinking as well as programming.

Introduction

Computer programming is a basic course for college students of science and engineering majors, it aims to guide students to understand the ideas and methods of programming, and to master a language to solve problems by programming\(^1\). In recent years, with the propose and development of computational thinking\(^2\), it is becoming a trend to solve problems in various fields by means of computational methods and tools, thus it becomes more and more important of this course.

But in teaching practice, it is quite difficult to reach the objects of this course. Firstly, it needs great efforts on coding and debugging of programs, but Students are often afraid of tedious syntax of the language, and finally lost confidence for programming when meet program errors again and again. Secondly, the procedure of programming is indeed a procedure of logical thinking, but for many fresh students, the ability of logical thinking is still weak, there is some difficulty for them to understand the logic of programs, so we need a gradual training method for logical thinking. Thirdly, the flow diagram that been used for expressing the algorithm, but for it cannot be executed, there has no way to verify the result of the algorithm, we need stronger tools to express algorithms and verify them.

In recent years, a lot of visual programming tools came into being, for example, Scratch\(^3\), Blockly\(^4\), Snap\(^5\) and so on. By introducing of visual programming tools in computer programming course, the problems mentioned above can be solved effectively. First, for that the core of computational thinking is the ideas rather than the language syntax, by using of visual tools will simplify programming, thus students will focus on the program logic rather than the language syntax, it will help to overcome the fear of programming and cultivate there interesting. Second, visual program help to express the logic of algorithm clearly, thus it helps to train logical thinking through visual programming, lastly, by modeling with visual programming tools, it is easy to execute the algorithm immediately, thus convenient to verify the algorithm.

The Usage of Visual Programming in Teaching Practice

The author have practiced visual programming in three stage. First for algorithm design, made use of visual program to take place of flow diagram, intuitively introduce the logic of algorithms and verify them immediately. Second for coding, combined with the syntax and control flow of a programming language, transform the visual program into codes of the corresponding language. Third for modular design, create program modules by means of customized block, thus understand the ideas of decomposition and modular design for complex system. Next we will illustrate on the three topics.

Make Use of Visual Programming in Algorithm Design

Algorithm is the soul of program, and also be the key for developing computational Thinking and logical thinking. For American students, they usually begun visual programming in primary and secondary school, so that the logical thinking had been developed, but for Chinese students, they always lacking this ability. At the beginning of the course, we propose to use the simplest and most
interesting tools for algorithm design, for it help to attract interests of students, the author recommend to use Scratch from MIT Media Lab. Scratch provide an online web version as well as an offline installation version, both are easy to use. Although it is designed mainly for primary and secondary school students, it is also attractive for college students.

In algorithm design, according to the control logic of sequence, selective, loop as well as mixed structure, Select typical cases and implement them by scratch blocks respectively, thus demonstrate the algorithm intuitively.

For sequence programming, introduce the question of “chicken and rabbits in a cage”, to make students understand the process that transform from a question to a program, the common flow of “Input-Process-Output” for each program, and the basic concepts of variable, expression and so forth, as shown in Figure 1.

For selective programming, use the demo of “Input 3 numbers, find the maximum one and output the value” to show the basic flow of the algorithm, and then implement the algorithm by means of “if” and “if-else” blocks, as shown in Figure 2.

For loop programming, first demonstrate the program of “summation from 1 to 100”, as shown in Figure 3, to introduce the control logic of loop, meanwhile, by adding delay item in the block, it will be easy to watch the changes of variables, such as loop variable i and summation variable s, thus understand the logic of loop control clearly. Second, introduce the demo of “calculate π by the following formula: $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + ......$”, as shown in Figure 4, it will help students to understand
the idea of iteration, moreover, by changing the number of iterates, the computation load and accuracy can be observed easily.

Through these examples, we can find that by using of visual tools instead of flow diagram, algorithm can be described Simply and intuitively, furthermore, the algorithm can be executed and verified immediately, thus it will help students to understand algorithm easily.

Make Use of Visual Programming in Coding

It is easy to find that the blocks in visual programming have great similarity with the statements in a language, once learned the syntax and control flow of a language, it will be easy to transform the visual program into the coding of a given language. Take the algorithm of Figure 2 and Figure 4 as example, the generated code is shown as following.

```c
int a, b, c, max ;
printf("Input a, b, c please : ");
scanf("%d %d %d", &a, &b, &c);
if(a > b) max = a;
else max = b;
if(max < c) max = c;
printf("Max = \%d", max);
```

```c
int s = 1, d = 1 ;
double sum = 1, t ;
do { s = -s; d += 2;
t = (double)s/d ;
sum += t;
} while (d < 10000);
printf("PI = \%lf", sum*4) ;
```

Students can be guild to programming by 2 steps: first, leading by the teacher, to design visual programs, so as to understand the algorithm logic, and then, hands-on by students, contrast with the syntax and control flow of the given language, transform the visual program into codes of the language. When designing visual programs, it also be nature to introduce concepts of variable, operator, expression, control flow, input and output and so forth, and these concepts can be easily mapped into syntax of a given language. By separating of the two steps, the original complex program can be divided into to 2 procedures that is relatively simple, thus help students to overcome the fear of difficulty, and help to catch the key issues of programming.

Make Use of Visual Programming in Modular Design

Modular design is one of the most important content of computer programming, and function is the key block for modular design in high-level language. For the simplicity of Scratch, it is hard to simulate function by visual blocks, thus we need a stronger visual programming tool. Blockly, which is produced by Google, is an alternative choice, it provides a lib of web based components, that is convenient to build a visual programming editor, it also support the conversion of visual programs into codes of JavaScript, Python and some other languages. For the powerful of it’s components lib, there have been a lot of educational software developed by Blockly, for example, App Inventor, etc. But, Blockly is actually developed for developers, users are required to have fundamental knowledge of web development, with skills of web environment configuration and web program re-development, unfortunately, few of teacher or student have these skills. In teaching practice, we need a Blockly based application, or some other visual programming tools like the application, the author recommend Snap that produced by UC Berkeley.

Snap provide a web based visual programming environment that is easy to use, the program can be saved at both client machine or cloud side so as to be opened next time, it has fully support for modular design, and even support to create new control structures, thus it can be used for course of programming perfectly.

Snap create program modules by customized blocks, for example, to find the maximum value in a list of data, we can design a block as shown in Figure 5.

In this block, the first line defined the prototype of the function, with the function name getArrMax, an array variable “s” as input parameter, and a return variable “m” as the maximum value. The core algorithm is defined in the middle part, firstly grant the first item of the array as the maximum value,
and save it to the variable \( m \), then iterate through all items of the array, when finding an item that is larger than \( m \), set the item value to \( m \), thus, when finish the loop, the value in \( m \) will be the maximum value of the array. The last line will return value for the main program.

![Figure 5. Block to find the max value of array.](image)

To demonstrate nested calls of functions, an example of finding combination number \( \binom{n}{m} \) is designed, with the formula of \( \binom{n}{m} = \frac{m!}{n!(m-n)!} \). First define the algorithm that calculate \( \binom{n}{m} \) as a function, and then, for that the calculation of factorial was called repeatedly, it is reasonable to define another function for it. Thus the first function was called in main program, and the second function was called in the first function, the nested relation was revealed clearly, the visual programs was shown in Figure 6a and 6b.

![Figure 6a. Block to calculate \( \binom{n}{m} \).](image)

![Figure 6b. Block to calculate \( n! \).](image)

Recursion is a unique thinking in compute science, it means that a function can be called by itself directly or indirectly. To show recursion to students intuitively, first designed the following algorithm to draw a tree recursively, as shown in Figure 8. Obviously the tree consists of 6 layers, in the first 5 layers, each branch can be treat as a sub-tree with 2 sub-branch, up to the last layer, there will be only leaf, without any sub-branch. Thus define the recursive form function “tree”, in which, first decide whether it is in the last layer, if true, exit without any action, otherwise, draw the sub-tree by 3 steps: draw the trunk of current layer, draw the left sub-tree, and draw the right sub-tree, then finally, reset the paint-brush to the initial position.
Similarly, the factorial function can be redesigned in a recursive way, with the following formula:

\[ n! = \begin{cases} 
1 & n = 1 \\
n \cdot (n-1)! & n > 1 
\end{cases} \]

Thus the new factorial function is shown in Figure 8.

**Conclusion**

In this paper, the author introduced visual programming into teaching practice of computer programming course, mainly studied on the usage of visual programming in algorithm design, coding as well as modular design. Practice shows that the following improvement was achieved: Firstly, improved the student’s interests for programming, thus no longer afraid for it, secondly, the ability of logic thinking was developed, therefore, they can skillfully describe algorithm and verify it, thirdly, the ability of reading and writing codes was improved significantly. In next research, we will focus on the cultivation of computational thinking, introduce more typical algorithms that reflect ideas of computational thinking, combine to use visual tools and language to implement these algorithms, guild students to experience the unique thinking of computer science, and learn how to solve practical problems by means of computational methods.

**References**

[1] Xiaojun Bai, Shujuan Huang, Research on computer fundamental education based on computational thinking, International conference on social science, education and Humanities research( ICSEHR), 2017, V152.


[5] Information on https://snap.berkeley.edu/