

Cultivation of Students' Computational Thinking under the New Curriculum

-- App Inventor-based Intelligent Recognition App Design and Development

Ming Li *

Key Laboratory of Educational informatization for Nationalities Yunnan Normal University,
Kunming, China

* Corresponding Author Email: 1964955437@qq.com

Abstract. In April 2022, the Compulsory Education Information Technology Curriculum Standards (2022 Edition) were introduced, and the curriculum concept mentions "selecting curriculum content that emphasises both scientific principles and practical applications" and "advocating authentic learning" to focus on cultivating students' integration of theory and practice. It also explicitly focuses on the development of students' computational thinking. An action research approach is used to design a multifunctional intelligent recognition software and to learn and design with learners to gain a better understanding of the App Inventor platform. Through the design of this software, the experience of designing software development based on the App Inventor platform is summarised on the one hand, and how to improve students' computational thinking in the design process on the other.

Keywords: App Inventor; Intelligent Recognition; Computational Thinking; New Curriculum.

1. Introduction

In the Education Information Technology 2.0 Action Plan, the Ministry of Education proposed to "enrich the content of the artificial intelligence and programming curriculum to meet the development needs of the information age and the intelligent age." [1] In May 2022, the Ministry of Education released the New Curriculum for Compulsory Education - Information Technology 2022, in which it is clearly stated that data, algorithms, networking, information processing, information security and artificial intelligence are the logical lines of the curriculum. [1] In May 2022, the Ministry of Education released the "New Curriculum for Compulsory Education 2022 - Information Technology Curriculum", in which it is clearly stated that data, algorithms, networks, information processing, information security, and artificial intelligence are the logical main lines of the curriculum, and that "computational thinking" is the objective of the IT curriculum: to apply the ideas and methods of the computer science field, and to be able to abstract, decompose, and model problems. The course aims to apply the ideas and methods of the computer science field, to abstract, decompose and model problems, and to design algorithms to form solutions. [2] With the emergence of NetEase's programming cats and makerspaces, China is paying more and more attention to programming education, following the pace of international competition, not only from the perspective of the learner, but also building teachers with high information literacy, and exploring new educational models of programming education, such as steam education and project-based education models. The emergence of these new models and teaching methods has helped to strengthen the foundations of learners' development and enable them to move forward in the wave of information technology.

Nowadays, IT classroom education mainly teaches the basic operation of office software and some basic knowledge about computers, and the main languages that learners are exposed to are LOGO language and BSAIC language, which are complex and difficult to understand and difficult to mobilise learners' interest. In a survey conducted by the Ministry of Education in 2012 on the "most desired modules in the IT curriculum", the unexpected exposure was that the module "Preliminary Artificial Intelligence" had reached 33%. [3]

App Inventor is a visual programming platform that encapsulates code, is easy to use, can be integrated with education, has a patchwork of code, is easy to work with, etc., and greatly reduces the

barrier to learning programming. It allows learners to learn about algorithms in the process of building blocks, so that they can quickly turn their imagination into reality and take the lead in identifying and solving problems while learning how to design the software. For learners who are tired of cumbersome code, the traditional teaching of programming by teachers did not allow for a good visualisation of what programming meant, creating a situation where it was difficult to understand what they were hearing, but difficult to do in practice. Now the platform allows learners to design their own software with ease.

2. Computational thinking and App Inventor

2.1 Computational thinking-related research

The emphasis on computational thinking in foreign countries, particularly in the US with the CPATH programme in 2007 and the CDI programme in 2008, has been on the integration of computational thinking with other disciplines, particularly mathematics. In 2012, Professor Yasmin B. Kafai proposed the following new typical steps for developing students' computational thinking: (1) write code to create applications; (2) apply improve life; and (3) modify the application to recreate it [4].

Looking at domestic research, the more accepted definition in the industry is given by Professor Yi-Zhen Zhou in his article "Computational Thinking" published in the prestigious journal Communications of the ACM: computational thinking refers to the use of basic concepts of computer science Computational thinking is a range of thinking activities that cover the breadth of computer science, such as problem solving, system design, and understanding of human behavior. [4] In the new curriculum, four stages are used to explain the goals that students should achieve in computational thinking at the compulsory education level. This is shown in Figure 1. It is easy to see that today's information technology requires students to use computational thinking to solve some basic problems. The release of the 2018 Education Information Technology 2.0 Project, which emphasises the deep integration of information technology with the subject, and how to improve students' computational thinking in this integration process is something we should think about.

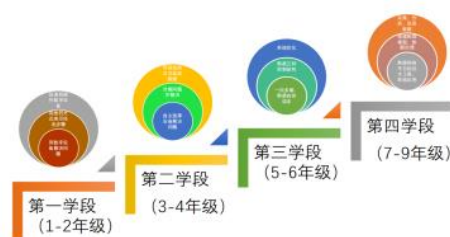


Figure 1. What students in compulsory education should be aiming for in terms of computational thinking

2.2 App Inventor Introduction

App Inventor, which means "application inventor" in Chinese, is a graphical programming tool that allows you to build modules online and convert complex programming language code into various components, developed by Google. The App Inventor platform consists of two main components: component design and logic design.

The component design interface is made up of four parts: the component formation panel, the work panel, the component list and the component properties. This is shown in Figure 2.

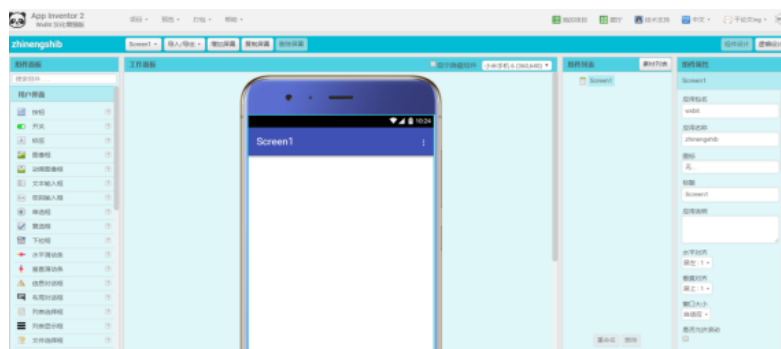


Figure 2. Component design interface

Logic Design, which is the program code editing window in App Inventor, consists of modules and working panels. This is shown in Figure 3.

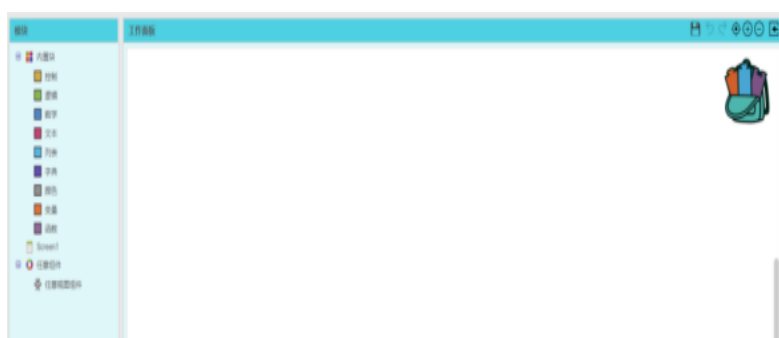


Figure 3. Program code editing window

2.3 Features

Professor Guo Shouchao introduced App Inventor with the following features: (1) easy to match the development environment (2) the development process is simple and easy to operate (3) it does not require much programming knowledge (4) the development cycle is short (5) App Inventor supports offline development [5].

2.3.1 Convenient development environment

The latest version of App Inventor mentioned above was developed by MIT, and there are similar ones in our country, such as App Inventor 2WxBit Chinese Enhanced Edition (<https://App.wxbit.com>) developed by a team from South China University of Technology, and the Guangzhou Education Information Centre (TEI) erected App Inventor2 Server (<http://App.gzjkw.net>) both of which are designed online, and the desktop version of AppInventorDesktop2019 designed offline users develop on these platforms without the need to build a complex environment.

2.3.2 Simple development process

Users are able to set up a layout design, in the component design layout, which can be done by dragging and dropping components through visual graphics modules to complete a layout design of the software, and in the logic design layout, where users can drag modules into the working layout and code them to implement different algorithms. Users do not need to master complex Java, Django, C# and other high-level programming languages.

2.3.3 Short development cycle

As a WYSIWYG programming tool, and as its project leader Harol Abelson (MIT professor) says, App Inventor writes applications that may not be perfect but they are accessible to the average person and can often be completed in minutes [6]. This allows teachers to teach task-based or project-based teaching in the classroom. Furthermore, according to previous research, primary and secondary learners are able to sustain their attention in the classroom for approximately twenty minutes, which makes the short time frame even more effective.

2.3.4 Reduced developmental errors

The match between "blocks" and "blocks" is completely fixed, so that if there is a grammatical error, the two blocks cannot be joined together, and if there is a small error, it is prompted in the bottom right-hand corner, so that the threshold of the curriculum design course is greatly reduced. The time spent correcting errors at a later stage is reduced and therefore the development workload can be reduced.

2.3.5 Development and timely debugging

Users can debug the project by downloading and installing AI Companion on the mobile phone and by packaging the development project as an apk and displaying it to the mobile phone. In particular, it is convenient and efficient to download and install AI Companion on the mobile phone for debugging, so that you can instantly check the running status of the app and make changes to some of the content.

3. Design of App Inventor platform for intelligent recognition based on the development of computational thinking

This section details the design and development of the App Inventor intelligent recognition app, based on which I have broken down the software design process into sections and which aspects of computational thinking can be achieved and developed in these stages under the new curriculum. This is shown in Table 1 below.

Table 1. Functional refinement

阶段	阶段实现的目标	计算思维的培 养	新课标计算思维目标
第一阶段 (课前预习)	1.学习者能够通过视频学习生活中有哪些主要的识别种类。 2.能够熟悉使用“微信”等APP的识别软件。	问题的提出	1.通过生活中的实例，了解算法的特征和效率。能用自然语言、流程图等方式描述算法。(第三阶段)
第二阶段 (前期准备)	1.学习者能够独立注册登录App Inventor 账号； 2.学习者熟悉App Inventor 工作界面。 2.学习能够独立注册百度云(数据库)的账号获取APIKEY等。	推理思维、 抽象思维、 系统思维	3.知道网络中信息编码、传输和呈现的原理。能通过软件与硬件相结合的项目活动采集、分析和呈现数据。(第四阶段)
第三阶段 (基础设计阶段)	1.了解App Inventor的开发流程； 2.能够在组件设计界面设计屏幕的相关组件； 3.学习网站简单参考案例，能够设计简单语音录制与播放等。	抽象思维、算法思维、系统思维、	2.对于给定的任务，能将其分解为一系列的步骤，使用顺序、分支、循环三种基本控制结构简单描述实施过程，通过编程验证该过程。(第三阶段)
第四阶段 (核心设计阶段)	1.能够编写“登录”界面的程序； 2.能够编写“文字识别”的程序； 3.能够编写“图像识别”的程序； 4.能够编写“语音合成与转换”的程序；	抽象思维、算法思维、系统思维、	1.在实践应用中，熟悉网络平台中的技术工具、软件系统的功能与应用。 2.能根据需求，设计和搭建简单的物联系统原型，体验其中数据处理和应用的方法与过程。(第四阶段)
第五阶段 (优化设计阶段)	1.学习者优化系统的界面的设置； 2.学习者优化系统的程序。	抽象思维、算法思维、系统思维、优化思维、纠错思维	4.通过案例分析，理解人工智能。根据学习与生活需要，合理选用人工智能，比较使用人工智能和不使用人工智能处理同类问题效果的异同。(第四阶段)

3.1 Login screen design

The login screen, which is designed to be more user-friendly and in line with current smartphone functionality, makes use of face registration and account registration as well as fingerprint authorisation for login. The main focus of this design module is to develop the learner's ability to

abstract in their computational thinking and to recreate the login scenario in their mind during the design process. The figure below shows the main interface for 'Login'. The core code for face recognition login is shown in Figure 4 and the core code is shown in Figure 5.



Figure 4. Main login screen

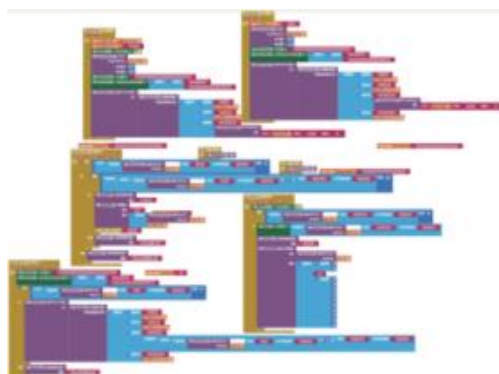


Figure 5. Core code for login screen

3.2 Image recognition design

The design of the main function involves four sub-categories, namely "plant and animal recognition", "wine dish recognition", "object-related data recognition" and "vehicle recognition", "Vehicle recognition", allowing users to easily select the type of recognition they need. In the image recognition design, I take "plant and animal recognition" as an example to describe the system process in detail, other types of recognition in a similar way.

3.2.1 Plant and animal recognition

The module can recognise photos of plants and animals from the mobile phone gallery or photographs, and the results of the recognition will be displayed at the bottom of the phone and the credibility of the recognition, so that plants and animals that have not been seen in the field or elsewhere can be recognised at any time to understand their knowledge.

The demand side of the system uses the data interface provided by the data service provider (the system uses the Baidu cloud database) to obtain the required data. The platform processes the data to ensure that it meets the needs of the user. The API interface for the entire image recognition function is the Baidu Cloud Database. The Baidu Cloud Database uses simple JSON format data, which is designed to be used by people who are new to the platform. Users can register on the Baidu Cloud platform and apply for an APPKEY to use the data interface to obtain data. The link to call the Baidu Cloud API is (http://aip.baidubce.com/rest/2.0/image-classify/v1/plant?access_token=) The detailed logic design module is shown in Figure 6



Figure 6. Calling the core code

When the Baidu Cloud Data Service API returns data, the JSON data is parsed to obtain the values of the relevant keywords and displayed on the screen. To find the values corresponding to the keywords "name", "score" and "log_id" in the JSON return text, we can use the list component provided in "Key list in the list component. The "not found, return" method is used to do this. A multi-layer search is conducted to find the keywords "name", "score" and "log_id" to obtain the corresponding values and assign them to the tags to display the data. The name of the plant is interpreted by the user according to the level of confidence, and then verified by a search in the mobile browser, as shown in Figure 7.



Figure 7. Retrieving the core code

When the "Start Identifying" button is clicked, the web client's URL sends data to the Baidu Cloud Platform API to request the data service. If the data is obtained, the result will be returned.

Summary: The difficulty in this design lies in the call to the database, and in the logical design, there are a lot of algorithmic applications, such as the use of "selection loops" in the "query results" section, which develops the algorithmic ability of the learner's computational thinking, and as mentioned above. As mentioned above, the design of the 'subcategories' of image recognition is similar, and the teacher can explain this through an example so that learners can discuss and imitate the flow chart, which is automatically generated through algorithmic thinking, and the data is processed and presented in a highly logical way.

3.3 Text recognition

In today's work and study, we often encounter some in the case of no electronic version of the document and want to edit the text of paper documents or extract some text on the photo, the mall on the text recognition software a lot, but some are limited, some charge, always not so satisfactory, the module is designed to allow us to picture the text can "move". The module is designed to allow the text on our images to "move". Here we will only look at generic text recognition. This module is designed to add a text copying function that allows the user to copy text and edit it very easily. The core logic of the design module is shown in Figure 8.

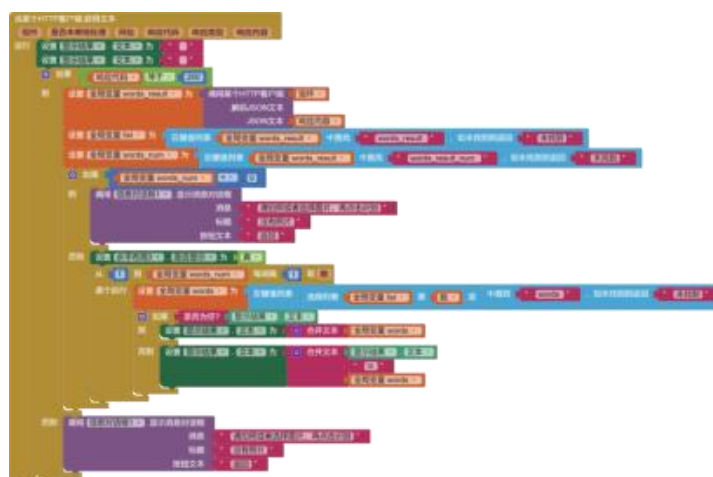


Figure 8. Text recognition core code

In the same way, in word recognition, there is a process whereby the identified words can be reproduced, reflecting a holistic approach and developing the ability to systematise in computational thinking.

3.4 Speech synthesis and conversion

3.4.1 Speech reading

This function converts the text entered by the user into speech, using a text-to-speech converter in the component design interface, which reads the text entered by clicking on the corresponding button.

3.4.2 Voice Recording

This module records the user's voice or relevant audio into a database, which can then be played back. When the recorder has finished recording the voice, it will encode the collected voice and store it in a specified directory in the database, then click on the analog button and the audio player will call up the audio inside and play it. The logic design for starting recording and playing the recording is shown in Figure 9.



Figure 9. Voice reading code

3.4.3 Voice conversion and text copying

This module allows the user to record their own voice and then click on Convert, the recognised text will then appear in the text box at the bottom of the screen, a Copy and Empty button will pop up, if the user needs the text, click on Copy, if they do not need the text they can click on the Empty Text button to clear it. This module uses the components of Baidu Speech Recognition and TaifunClipboard. The logical design of the speech recognition, text clear and copy buttons and the speech recognition component is shown in Figure 10.



Figure 10. Core code

So far all the core functions of the software have been designed, the next major work is to optimise the design, not only to optimise the code, but also to beautify the interface of the software, so that users can have a better feeling in the use. In summary, the "intelligent identification" software has been completely designed, the software is not only in the functional design, or in the interface design firmly grasp the "intelligent" word, mentioned above "text recognition", "image recognition", "voice recognition", "translation" these four functions, the specific use of the author in chapter 3 has been briefly described. "Learners should reflect their own personal ideas in imitating the reference, and not be limited to the author's design.

4. App Inventor project to develop learners' computational thinking under the new curriculum

Computational thinking is a fusion of mathematical, scientific and engineering thinking, as evidenced by researchers' studies of computational thinking. It requires mathematics as the basis for

the scientific design and implementation of engineering [8], but computational thinking does not simply emphasize mathematical computational thinking, logical relationships, etc., but rather emphasizes how to combine theory and practice in problem solving. In this way, the author will explain what computational thinking is fostered in the software development process in the following points. The video material that precedes the teaching of flipped instruction is what allows students to understand natural algorithms and to know that some problems have multiple solutions (using mathematical problems as examples). In traditional programming instruction, methods such as sorting, recursion, and exhaustive enumeration are generally used to solve problems, and these methods are all reflections of computational thinking in the design process, which is of course applied at all times during the design process described above, as shown in Figure 11 above, in which the core code block makes use of call statements as well as the process of judging, calling. In this core block of code, the process of calling statements and judgements is used, calling the "Baidu cloud database" and "judgements" to identify the trustworthiness of the results, and through the use of calling statements and judgements, learners are better able to exercise their awareness of calling and judgement statements, and to relate some of the traditional knowledge they have learnt about procedures. The use of call and judgement statements better exercises the learners' awareness of call and judgement statements and enables them to link some of the traditional knowledge they have learnt about programs, reflecting the abstract thinking skills of computational thinking. Students are able to draw software development flowcharts for tasks given by the teacher in the classroom in a task-driven and project-based learning mode, and to master the three main control structures of sequence, branch and loop in the design process. This is the third stage of the development of computational thinking in the new curriculum. In 'image recognition', as shown in the diagram below, the code block in this section retrieves all the keywords in the data returned from the call to the database in the displayed results. The learners can find and display the results of all queries in the interface. The use of exhaustive enumeration is a good way to develop the idea of exhaustive enumeration. In Speech Recognition we have involved the components 'recorder' and 'audio player'. The recorder transfers the audio to the audio player after recording. This is a process that demonstrates the nested nature of the program, with two components that are both independent of each other and interconnected. After the initial design of the software has been completed, students need to optimise the software and understand the importance of feedback in optimising the software. Throughout the design of the software, students are able to complete their tasks while also mastering the use of relevant technical tools and being able to independently build some models, know the flow of data calls, etc. They are also able to know how the data is presented through the recognition function, as the "intelligent recognition" software is needed in life, for later students who are interested in. If students are interested in computers they will try to learn the front-end and back-end themselves just to improve the accuracy of recognition. These are the goals that teachers are required to achieve for students in Key Stage 4 under the new curriculum.

This paper designs a model for developing learners' computational thinking based on the APP Inventor project based on the design of the whole process as shown in Figure 11: starting from the platform with a project learning teaching method, asking questions, teachers regenerating the links to guide students in data analysis, choosing whether to "student independent inquiry" or "collaborative exchange" depending on the size of the project. The teacher guides the students to analyse the data in the next session, and depending on the size of the project, to choose between "student independent inquiry" or "cooperative exchange" to break down the problem, to design a solution to the problem and then to break it down one by one, and then to design a product to verify whether it meets the needs, and after meeting the basic needs, the teacher guides the students to deepen the difficulty to upgrade the functions of the software. In the process of designing the "intelligent recognition" software, students can upgrade to animal recognition and dish recognition after they have completed the recognition of image recognition. Throughout the model the teacher should always guide students around problem solving to complete the design of the project. In the assessment stage teachers can

use a variety of assessment methods such as 'student performance demonstration type' and 'stitching assessment'.

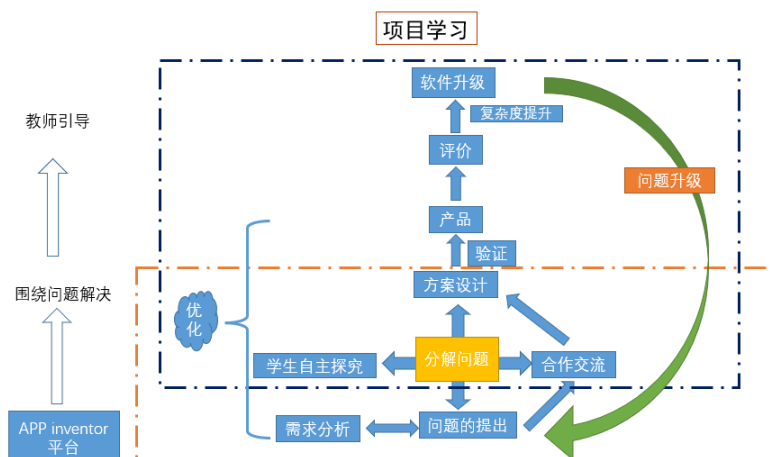


Figure 11. Model for developing learners' computational thinking based on the APP Inventor project

Through the design and development of the "Intelligent Recognition" App, the following are some of the author's recommendations for design and development and the development of computational thinking:

4.1 Modular design

In the process of designing the Smart Recognition App above, we can easily see that the outstanding advantage of App Inventor is that the easy operation and the visual effect of the modules enrich the perception of the learners and greatly enhance their interest in learning, encouraging them to actively learn and explore. As we know, learning to create software is essentially a process of problem solving and problem solving. We take the above example of "text recognition", where we first propose a problem such as "editing text on text", and modularity teaches us that the next design development is based around such a demand problem, in which learners go through the process of design, algorithms. These thinking tools enable learners to realise their own creativity and gain a rich learning experience.

4.2 Clever introduction of flowcharts

The clever introduction of flowcharts in APP inventor is essential, as learners can see that drawing flowcharts on the way can make the process of designing a program logically clear and greatly improve learners' learning efficiency, and know what kind of steps the program is designed to follow in this lesson. Function design, for example, in the design is we first achieve "animal recognition" function, according to the flow chart, you can set the first step: ask the question: how to achieve the function; step two: in the component interface according to their own design interface for component building; step three: the programming of the writing, in the logic design interface, the visual statement Step 4: As the functional design of "animal identification" is different from the functional design of other sub-categories, it is possible to extend the solution. In such a process, abstraction and decomposition methods are used to dissect complex problems, allowing learners to build a general model of the structure and to develop the ability to think computationally in terms of abstraction, algorithms and optimisation.

4.3 Decompose important and difficult points to guide learners to break through independently

In the above example, when designing the image recognition function, learners were asked to make bold guesses on the basis of the implementation of the text recognition function. In the course of the lesson, the teacher asks a big question, for example, what information is available in the result of the

image recognition? The modular approach to system functionality is used to find the "cure" for each module's problem, to explore the relationships between the modules, and then to facilitate the learners' own verification of the final results so that they can gain a deeper understanding of the problem.

4.4 Setting objectives and building learning models

By using the rich programming content of APP Inventor, teachers can use the above examples of visualisation to improve today's teaching format, establish class-specific learning models, and teach learners to improve their computational thinking skills in everyday life. In the programming of the Intelligent Recognition App, the learners have already learnt simple programming (programming cats and other children's programming) and have a good understanding of the general framework, but are often a bit inferior in the details and lack the concept of practical application.

In the design process, the theoretical model is used as a premise to design functions such as text recognition, so that learners have a detailed knowledge of how to call the database and so on.

4.5 Selecting appropriate content to promote interest in learning

Teachers should start from the actual situation of learners, grasp the learning characteristics of learners, write corresponding lesson plans, enhance learners' interest in learning, and ultimately achieve good results. Teachers can also consider the integration of subjects, not limited to the subject of information technology, for example, you can let learners create "rubbish classification" App, "idiom dictionary" App, "four operations" App and so on, which can let learners consolidate the knowledge of other subjects.

This will allow learners to consolidate their knowledge in other subjects, killing two birds with one stone. A new classroom is created in which the learner is the main subject and the teacher is the leader.

4.6 Focus on process

The development of computational thinking is not simply a matter of building blocks on the APP Inventor platform, but of experiencing the process of writing code. Computation is "the process of executing an algorithm that demonstrates both the methods of data analysis and the motivation of inner thought and discovery" [9]. The author argues that the basic idea of computational thinking may be lost if one simply seeks to write code to get results. Algorithm design and programming are used extensively throughout the design process. Losing these would not allow students to exercise the ability to design good algorithms to solve problems. We need to tell students why we are designing the way we are designing, rather than simply asking learners to follow the steps and 'follow the gourd'. It is more important that learners understand the problem-solving strategies that will facilitate the development of computational thinking.

4.7 Multidisciplinary integration and co-cultivation

Although computational thinking contains the word 'computing' and is often thought of as the exclusive voice of information technology, the author believes that the development of computational thinking should not be limited to App Inventor, but can be extended to Scratch, Alice, Daisy, Code Monster, etc. These visual programming can also be educated with a variety of blends like steam education. Every educator has the responsibility and obligation to develop computational thinking in learners, and this can be done through teaching activities, classroom content and so on, as it is after all a comprehensive approach.

5. Conclusion

App Inventor is just one of the suitable platforms to enhance learners' interest in learning programming. The software in this paper achieves the functions described above without the complex building blocks of programming, reflecting the rapid development of artificial intelligence technology

today. And nowadays, driven by the information technology era, the state also attaches importance to IT education. The introduction of several policies such as the Compulsory Education IT Curriculum Standards (2022 Edition) and the Education Informatization 2.0 Action Plan all show that there is still a long way to go in IT education and many more areas to explore. We need to be clear that the cultivation of computational thinking is a joint effort of researchers, educators and learners, exploring and practicing together. Researchers propose models, educators implement practices, learners give feedback and researchers improve, such a cyclical process, until a new model of computational thinking cultivation is created that suits our basic national conditions and the physical and mental development of our youth.

References

- [1] Notice of the Ministry of Education on the issuance of the Action Plan of Education Informatization 2.0 [J]. Bulletin of the Ministry of Education of the People's Republic of China, 2018(04):118-125.
- [2] Ministry of Education of the People's Republic of China. Compulsory education information technology curriculum standards: 2022 edition [M]. Beijing:Beijing Normal University Press, 2022.
- [3] Ren Youqun, Huang Ronghuai. Revision notes of high school information technology curriculum standards Revision group of high school information technology curriculum standards [J]. China's electrification education, 2016(12):1-3.
- [4] Yasmin B.Kafai & Quinn Burke. The social turn in K-12 programming: moving from computational thinking to computational participation [DB/OL]. <http://dl.acm.org/citation.cfm?id=2445373>. 2013-12-12.
- [5] Guo Shouchao, Zhou R, Deng Changmei, Di Changyan, Zhou Qingguo. Research on IT classroom teaching based on App Inventor and computational thinking[J]. China's e-learning, 2014(03):91-96.
- [6] Niu Xiao, Qin Jian. Design and debugging of App Inventor-based speech recognition and text translator app[J]. Computer Knowledge and Technology, 2020, 16(22):20-22+25.
- [7] Xu Chao. Research on the application of App Inventor-based computational thinking skills of high school students[D]. Yunnan Normal University, 2017.
- [8] Benson, D. Curriculum visions: The Australian Curriculum, Assessment, and Reporting Authority (ACARA) and Dwayne Huebner discuss civics and citizenship[J]. International Journal of Christianity & Education, 2015, 19(1):38-56.
- [9] Xu Chao. Research on the application of App Inventor-based computational thinking skills for high school students[D]. Yunnan Normal University, 2017.