"BASENGAJA" APPROACH - AN APPROACH OF TEACHING BASIC PROGRAMMING

Victor N. Papilaya
Electronic and Computer Engineering Faculty
Satya Wacana Christian University
Jl. Diponegoro 52-60, Salatiga 50711, Indonesia
e-mail: victor.papilaya@gmail.com

Abstract

"Basengaja" approach is an approach of teaching basic programming. This approach is designed to help the students who faced problems in following the lecture of basic programming. The basic idea of this approach is to reverse the sequences of regular teaching approach with two additional steps. The purpose is that in the such as mechanism the students will firstly learn the basic logic of thinking in programming instead of the complexity of language programming commands itself. Furthermore, this approach needs the simple IDE that can be used to simplify learning process. With this IDE, the students are not required to remember or to know commands of language programming to solve certain problem. For this reason, an IDE called Oledit is developed. Based on the direct observation and the profile of the students after learned basic programming using this approach, the result shows that this approach brings the positive contribution.

Keywords: Basengaja approach, basic programming teaching approach, Oledit

1. Introduction

The ability to develop a program is an important ability that need to be achieved by the students who study in Engineering Faculty. But achieving this ability is not easy for the students who really new in the programming world. There are many factors influenced this situation such as the spirit and motivation of students. The author has made informal discussion with these kind of students. Based on this discussion, the author has seen the main factor of problem they faced. There was no good connection between the sequences of teaching used and the initial knowledge owned by the students.

This problem can be explained using the Artificial Intelligent concept. The students are agents. Every agent has sensor and actuator, a sensor is used to receive perceptions from the environment and an actuator is used to generate responses. Every agent has also knowledge based (KB) and is actually the set of sentences [1]. Every sentence has its own syntax. When an agent receives a perception, this perception will be translated into sentence that can be used to make inference. The inference process is actually the process to check whether the sentence generated from perception is true in some or all models based on KB. The good inference can be made, if and only if an agent has enough axioms in its KB. If this is the case then, an agent can learn something from its perception. In our case, the perceptions are the informations given by lecturer. Based on our case explained above, the problem is that there are no enough axioms owned by the students. So learning process can not be done.

There are two solutions for this problem. The first solution is the students must improve their knowledge, in other words they have to increase the number of their axioms. The second solution is that, the perception must be changed. So this perception can be translated into sentence that can be used to make good inference. The “Basengaja” approach is a type of the second solution.

The “Basengaja” word was taken from the Ambon language which means the comfortable situation established by two persons during their talking. These two persons must have good relation, so they can choose the right sentences in order to create the comfortable situation.

2. The Concept of “Basengaja” Approach

Figure 1 shows the common sequences of teaching on the basic programming lecture. The lecture is generally started by explaining the language programming that will be used. The explanation contains the syntax and commands of the language programming. After that, the examples of source codes will be given and explained, and at the end, the output of these examples will be shown. Based on this mechanism, this approach is actually a type of the first solution explained above. Because the purpose is that, the students must firstly increase their axioms related to the language programming that will be used.

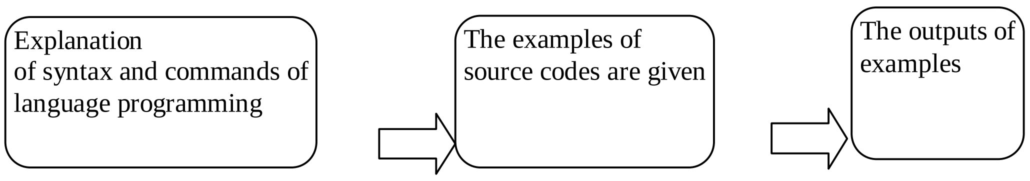
Explanation of syntax and commands of language programming
The examples of source codes are given
The outputs of examples

Figure 1. The common sequences of teaching on the basic programming lecture.

In the author's point of view, this approach is only suitable for certain type of students and can not be used generally. The reason is that, new students are generally have small numbers of axioms in the programming domain. So if the first perceptions received by students are syntax and commands of the language programming, then they will face difficulty to understand the lecture.

"Basengaja" approach reverses the sequences of teaching commonly used. This means, the output of the certain source code that will be explained becomes the starting point of explanation. In such as mechanism, the students can use their initial KB to understand the function of the commands which are used because they have already seen the behavior of the commands. If they have understood this function, the next step is that to explain the syntax of commands.

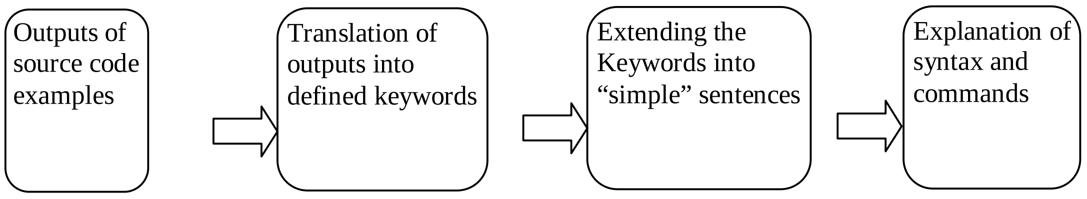
Outputs of source code examples
Translation of outputs into defined keywords
Extending the Keywords into "simple" sentences
Explanation of syntax and commands

Figure 2. The sequences of teaching based on "Basengaja" approach

The figure 2 shows the mechanism of "Basengaja" approach. The output of the source code example will be broken into defined keywords. These keywords have to be extended into "simple" sentence that can be easily translated into the source codes. "Simple" sentence means the sentence that can be easily understood and contains no commands of the language programming.

2.1 Translation of Output Into Defined Keywords

Following are some keywords used on the "Basengaja" Approach: print, input, repeat, and choose. Assuming that C/C++ language is used.

2.1.1 Print

This keyword is used to describe that a program has to print something on the screen. This keyword will be translated into command: printf or cout.

2.1.2 Input

This keyword is used to describe that a user has to give an input through keyboard. This keyword will be translated into command: scanf or cin.

2.1.3 Repeat

This keyword is used to describe that there is a looping process in a program. This keyword will be translated into command: for or while.

2.1.4 Choose

This keyword is used to describe that a user has to make decision. This keyword will be translated into command: if or switch.

For example, given the following output:

```
Input your name = victor
Halo victor, nice to meet you …
```

This output can be broken into keywords as follows:

Print
Input
Print

This process depicts the steps needed to write a program that can produce the given output.

2.2 Extending Keyword Into “Simple” Sentence

A simple sentence is a sentence that can be easily understood and contains no commands of the language programming. This sentence is required to easily to be translated into source code. This means, this sentence must describe the important parts of commands. Printf command, for example, can be used to print a message without any variable or a message with a variable contained user's input. The important part of this command is the existence of variable.

We will use the example given above. The "simple" sentences of the keywords generated from the given output are:

Print a message without variable
Input string value
Print a message with a string variable

If students can do transformation from the given output into "simple" sentences, actually they knew already the logic of thinking in the programming. This is the first important key that students must have before going into the next step in which they have to learn about the syntax and commands given in the certain language programming.

As mentioned before in the abstract section, "Basengaja" approach is also supported by an IDE. This IDE is used in order to make direct translation from the "simple" sentences into commands given in the language programming. This direct translation helps students to solve the given problem without any knowledge or minimum knowledge of language programming. The idea of this direct translation will be explained on the results and analysis section.

3. Research Method

The Research is done in the following steps:

1. Design an idea of "Basengaja" approach
2. Create an IDE that implements "Basengaja" approach. This IDE will be used as the supporting tool during the lecture.
3. Do lecture based on "Basengaja" approach. The students who attended this lecture are the students who had the problem in understanding the lecture of basic programming. The number of students in every lecture is limited. There were just four persons in every class. This configuration makes it possible to do direct oberservation on the level of understanding of every student.

4. Results and Analysis

The results and analysis will be explained based on the sequences of research method.

4.1 Designing an Idea of "Basengaja" Approach

The idea behind the "Basengaja" approach has been explained in the previous section.

4.2 Creating an IDE that implements "Basengaja" approach

An IDE called Oledit is the supporting tool of "Basengaja" approach and is developed using Gambas [2]. So this IDE is only exist on the Linux operating system. On its early version, this IDE is implemented as an editor of Openlib library. The detail information about this library and its implementation has been explained in the C/C++ goes to open source book [3].

The extension of Oledit's capability to support "Basengaja" approach is done by providing additional menus. These menus represent the given keywords and "simple" sentences as explained above. When a user selects certain menu, the commands related to keywords and "simple" sentences will be appeard on their program's source codes. Some videos related to this explanation can be seen at http://cplusplusindonesia.blogspot.com/2011/03/berkenalan-dengan-oledit-ide-untuk.html.

4.2.1 List of Menus for Supporting "Basengaja" Approach

Here are additional menus existed on the current version of Oledit:

Variables declaration --->	intenger	
	--->	float
	--->	string
	--->	more than one integer
	--->	single dimension of integer array
	--->	double dimension of float array

Print --->		a message without variable		
	--->	a message with variable	-->	integer

```
                                    -->    float
                                    -->    char
                                    -->    string
                                    -->    combination between
                         integer and float
                                    -->    combination between
                                           integer, float and string
Input ---> integer
     ---> float
     ---> char
     ---> string

Choose --->   using switch
     ---> using if --> one condition
                  --> one condition with else
                  --> some conditions (cascade)
                  --> some conditions (cascade) with else
                  --> some conditions (without cascade)

Repeat ---> using Goto-label
     ---> using while        --> infinite loop
                             --> finite loop
     ---> using do-while --> infinite loop
               --> finite loop
     ---> using for    --> inifite loop
                             --> finite loop
```

4.2.2 Examples of Menu to Codes Translation

Here are some examples of codes generated when a user selects certain menu:

selected menu:

Variable declaration ---> integer

codes generated:

```
int datInt;
```

selected menu:

print ---> a message with variable ---> combinatio between integer and
float

codes generated:

```
printf("This is integer value : %d and float value : %f",[var integer],[var float]);
```

selected menu:

input ---> integer

codes generated:

```
scanf("%d",&[var integer]);
```

selected menu:

choose ---> using if ---> some conditions (cascade)

codes generated:

```
if (first condition==1)
{
  //This block is executed if the first condition is true

} else
if (second condition==1)
{
  //This block is executed if the second condition is true

} else
if (third condition==1)
{
  //This block is executed if the third condition is true

}
```

selected menu:

repeat --> using while --> infinite loop

codes generated:

```
while(1)
{
  //This block will always be repeated

}
```

4.3 Doing lecture based on "Basengaja" approach

The lecture based on proposed approach was done in small groups. There were 7 groups. Every group contained maximum 4 persons. This lecture was conducted directly by the author. During the lecture, the author oberseved the responses given by students. The results showed that the proposed approach is good to use for giving ideas of how to solve given problem using certain language programming. When the students already understood the logic of thinking then the common sequences of teaching could be used in order to give clearer explanations about the commands which are used.

5. Conclusion

In this paper the new approach of teaching basic computer called "Basengaja" approach has been presented. Based on the experiments, the combination between the common approach of teaching and the proposed approach can bring better understanding for the students who had problems in understanding basic programming. Furthermore, with the supporting of an IDE called Oledit, the students can learn two important things at once: the logic of thinking to create a program and the commands of the C language programming needed to implement that logic.

The positive contribution brought by the proposed teaching approach has proved:

1. The sequences of teaching basic programming must be organized based on the initial knowledge base of students.
2. There is a need to use the right supporting tool during the teaching process.

References

[1] Russel SJ, Norvig P., *“Artificial Intelligent A Modern Approach”*, Second Edition. USA: Prentice Hall. 2003.

[2] Papilaya VN, “Membangun Oledit, Editor Bahasa C/C++ di Linux”, AITI Jurnal Teknologi Informasi. 2006; 2(5): 114-123

[3] Papilaya VN, Prestilano J, “C/C++ goes to opensource Cara Membuat Pustaka Sendiri”, Jogjakarta: Gava Media. 2006.