

# Penguin Compiler Guide

Răzvan Diaconescu  
Department of Computer Science  
West University of Timișoara  
e-mail: [razvan.diaconescu04@e-uvv.ro](mailto:razvan.diaconescu04@e-uvv.ro)  
github: <https://github.com/Razvanlog>

June 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>How to set up the Penguin Compiler</b>	<b>3</b>
<b>3</b>	<b>How to use the Penguin Compiler</b>	<b>3</b>
3.1	Initializing Variables . . . . .	3
3.2	Reading Values from stdin . . . . .	3
3.3	Writing Variables to stdout . . . . .	3
3.4	Changing the Values of Variables . . . . .	4
3.5	Conditional Statements . . . . .	4
3.6	Looping Statements . . . . .	4
<b>4</b>	<b>Example of a Program</b>	<b>5</b>

# 1 Introduction

This guide documents how to use Penguin Compiler, as well as its features. This guide will correspond to (Penguin Compiler) version 1.0.0.

## 2 How to set up the Penguin Compiler

In order to set up the program, you will need a C++ compiler. The one I used to create the program is g++, which is from the family of GNU Compilers. version 13.1.0. After compiling the program, the Penguin Compiler will be ready to use.

## 3 How to use the Penguin Compiler

After the program is built, you can now input files with the file extension **.pin** for the compiler to compile and execute their code.

In the Penguin programming language we can initialize variables, attribute the integer result of an expression to a variable, read and print to standard input and output and use loop and conditional statements.

### 3.1 Initializing Variables

In order to initialize a variable, the user needs to use the keyword **int** followed by the name of the variable. In Penguin, variable names can contain small or big characters of the english language, integers or special characters such as `-`, `@`, `#` or `$`. After initializing a variable, the value stored in it will be 0.

```
1 int Variable_Example#
```

### 3.2 Reading Values from stdin

In order to read values from standard input, the keyword **read** needs to be used, followed by the name of the variable where the read value will be placed. The only supported data type currently in Penguin is **integers**, any other value inserted which is of a different type, i.e. strings of characters or real numbers, will cause the compiler to raise a runtime error and stop executing the rest of the code.

```
1 read variable_example$
```

### 3.3 Writing Variables to stdout

Writing variables to stdout can be achieved by using the keyword "print" followed by the variable's name.

```
1 print variable_example$
```

### 3.4 Changing the Values of Variables

The values of the variables can be changed after reading. Writing the name of the variable followed by the character `=` and an expression, the stored value in the variable will be equal to the result of the expression.

```
1 variable_example=(variable_example1+2)/variable_example3
```

### 3.5 Conditional Statements

In order to use a conditional statement, you need to use the keyword **if** followed by a boolean equation. A boolean equation is made up of two equations separated by a boolean operation such as `!=`, `==`, `<`, `<=`, `>` or `>=`. After the boolean equation, the statement should be followed by the symbol `{`, which will denote the start of the scope of the code which should be executed if the condition is true, this scope will end with the symbol `}` on a single line.

### 3.6 Looping Statements

The form of the looping statements is similar to that of conditional ones. In order to declare a loop, use the keyword **while** followed by a boolean equation and the symbol `{` used to indicate the beginning of a scope. After writing the code you wish to repeat, end the scope with the symbol `}`. During runtime, the loop will repeat until the boolean equation no longer yields true.

## 4 Example of a Program

In this section of the guide, I will show you how to compile the example program.

```
1 int variable_example1
2 int variable_example2
3 read variable_example1
4 while variable_example1!=0 {
5     variable_example1=variable_example1-1
6     variable_example2=variable_example2+5
7 }
8 if variable_example2==50 {
9     variable_example2=variable_example2-10
10 }
11 print variable_example2
```

Placing the code inside a file with the extension .pin, we just need to enter a terminal and call the Penguin Compiler with the name of the file. For this example, I used VS Code to write the program.

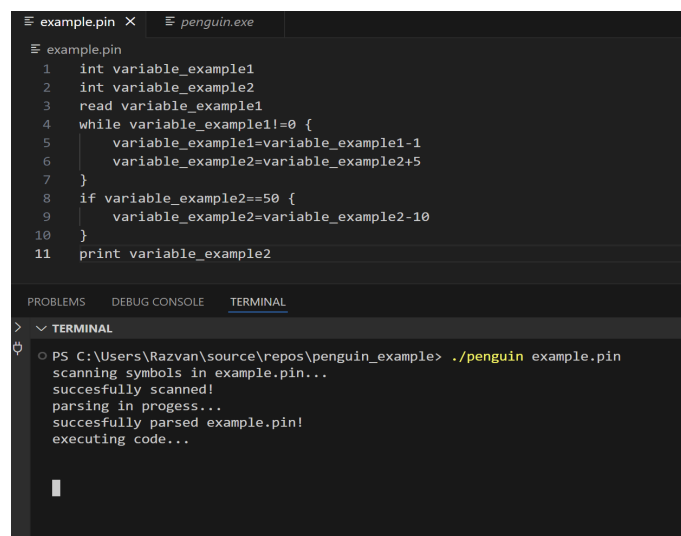
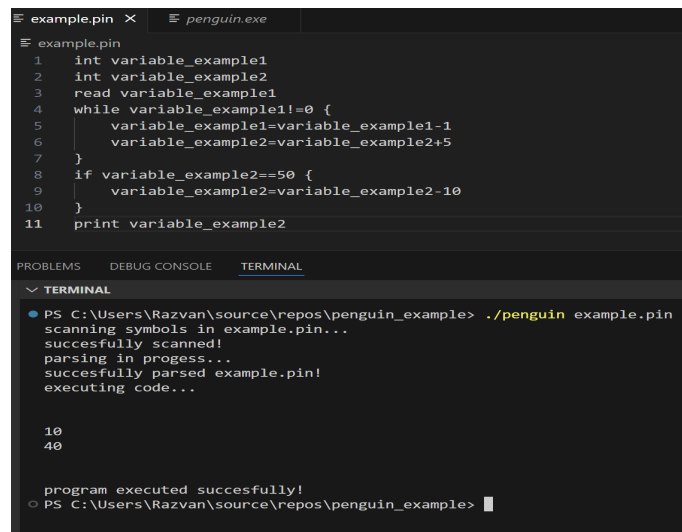


Figure 1: Screenshot after compiling the example.pin program

If the code of the program is written correctly it will begin executing the code. In our case the program will now wait for us to input a value.



```
example.pin x penguin.exe
example.pin
1  int variable_example1
2  int variable_example2
3  read variable_example1
4  while variable_example1!=0 {
5      variable_example1=variable_example1-1
6      variable_example2=variable_example2+5
7  }
8  if variable_example2==50 {
9      variable_example2=variable_example2-10
10 }
11 print variable_example2

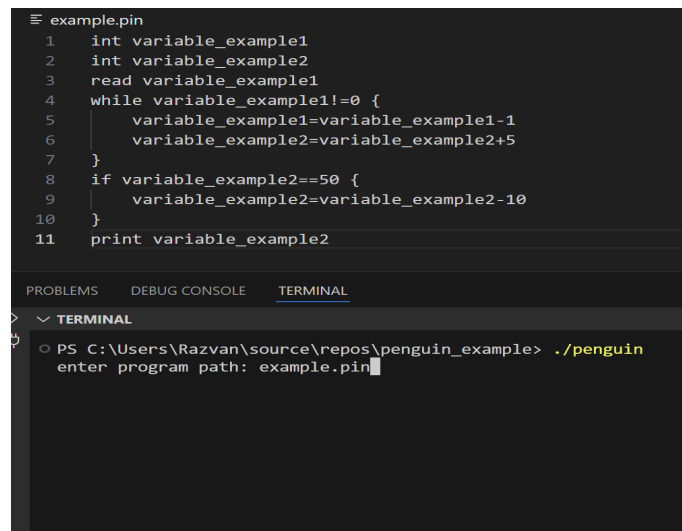
PROBLEMS  DEBUG CONSOLE  TERMINAL
TERMINAL
PS C:\Users\Razvan\source\repos\penguin_example> ./penguin example.pin
scanning symbols in example.pin...
successfully scanned!
parsing in progress...
successfully parsed example.pin!
executing code...

10
40

program executed succesfully!
PS C:\Users\Razvan\source\repos\penguin_example> 
```

Figure 2: The program after successfully executing

After we input the value, the program will continue on executing the code until the end.



```
example.pin
1  int variable_example1
2  int variable_example2
3  read variable_example1
4  while variable_example1!=0 {
5      variable_example1=variable_example1-1
6      variable_example2=variable_example2+5
7  }
8  if variable_example2==50 {
9      variable_example2=variable_example2-10
10 }
11 print variable_example2

PROBLEMS  DEBUG CONSOLE  TERMINAL
TERMINAL
PS C:\Users\Razvan\source\repos\penguin_example> ./penguin
enter program path: example.pin
```

Figure 3: The program after calling just the compiler

It is not necessary to enter the program's name while calling the compiler, you can call it inside the program.