Computing

Lesson 5: Pseudocode

Programming Part 3: Iteration

Rebecca Franks

Materials from the Teach Computing Curriculum created by the National Centre for Computing Education

Translate the pseudocode

Translate the pseudocode

Step 1

Use your knowledge of Python syntax to translate this pseudocode into Python code in Repl.it.

```
stored_pass <- "password"
password <- ""
pass_mismatch <- stored_pass != password

WHILE pass_mismatch
    OUTPUT "Enter your password:"
    password <- USERINPUT
    pass_mismatch <- stored_pass != password
END WHILE

OUTPUT "Access granted"
```

Translate the pseudocode

Step 2 Test your program using the example tests and output below.

Example

Note: Use this example to check your program. Given the input you see in this sample interaction, this is the output your program should produce.

The user is prompted to enter a password.	`Enter your password:`
The user enters a password	`Five`
The user is prompted to enter a password.	`Enter your password:`
The user enters a password	`123456`
The user is prompted to enter a password	`Enter your password:`
The user enters a password	`password`
The program ends	

Design a program using pseudocode

FizzBuzz

Introduction

During this activity you will be designing the algorithm for a FizzBuzz game. When writing your algorithm you can refer to the pseudocode (**oaknat.uk/comp-ks4-pseudocode**) document.

FizzBuzz: Scenario

FizzBuzz is a classic maths game where players shout out a sequence of numbers and replace them with Fizz or Buzz when they reach certain numbers. There are many different versions of this game. Your program should match the instructions given here.

Fizz	Divisible by 3
Buzz	Divisible by 5
FizzBuzz	Divisible by 3 and 5

The program should end when the sequence reaches 15. There is some sample output for the sequence on the next slide.

FizzBuzz: Scenario

Sequence	Output
1	1
2	2
3	Fizz
4	4
5	Buzz
6	Fizz
7	7
8	8
9	Fizz
10	Buzz
11	11
12	Fizz
13	13
14	14
15	FizzBuzz

FizzBuzz: Scenario

Your program will need to meet the following requirements:

- Start at 1 and finish at 15
- For each number in the sequence it should:
  - Calculate if it is divisible by 3
  - Calculate if it is divisible by 5
  - If it is divisible by 3 then it should display Fizz
  - If it is divisible by 5 then it should display Buzz
  - If it is divisible by 3 and 5 then it should only display FizzBuzz
  - Otherwise it should just display the number

Task: Design your program using pseudocode

Using the program requirements as a guide, design the algorithm for the FizzBuzz game.

Remember to write your algorithm using **pseudocode** not Python code.

Tip: If a number is divisible by 3 then it should be able to divide with 0 remainders.

Tip: look at your work from the trace tables lesson (lesson 2) to support you with some of the code.

Tip: Feel free to test little snippets of code in Repl.it as you go along

Task: Create your program in Python

Step 1

Using your pseudocode as a guide. Create the program in Python.

Step 2

Test that your program is working correctly by looking at the sample output in the scenario.

Tip 1: Remember that you now have several programs that you have created earlier that you can use to support you with this activity.

Tip 2: It is very possible that you might experience a logic error when you translate your pseudocode to Python code. Use a trace table to detect the error.

Optional extra task!

Once your code is working you can turn it into an actual game. Use the points below to extend your game if you wish.

Game requirements:

- Provide a welcome message
- Ask the user to enter a value
- Check it is correct against the stored value in the sequence
- If the value is correct it should move on to the next number
- If the value is incorrect then the game should be over