APS106 – Design Problem #4

Cryptanalysis

Friday March 4th, 2022

Agenda

Problem Background

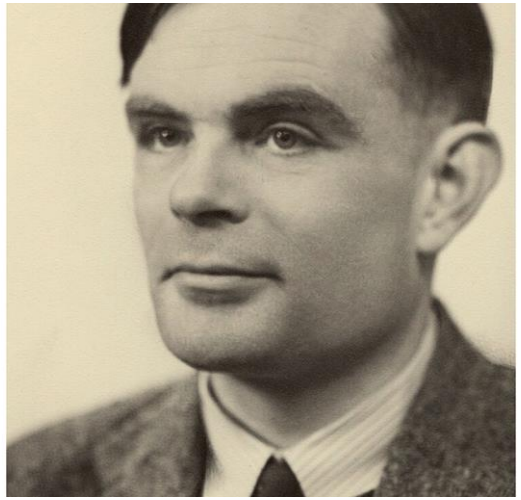
Learning Objectives

Coding

Cryptanalysis was a defining factor in WWII

Intersectionality of history and technology

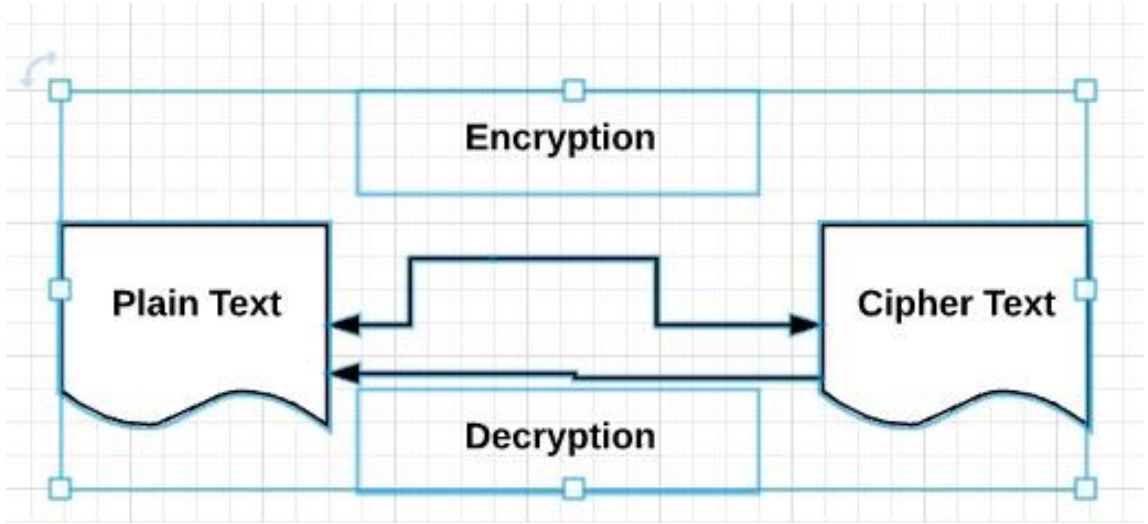
- Alan Turing
  - Pioneered the technology to decrypt Nazi Germany's secret communications during World War II
- These were called “Turing machines”



Alan Turing circa 1951

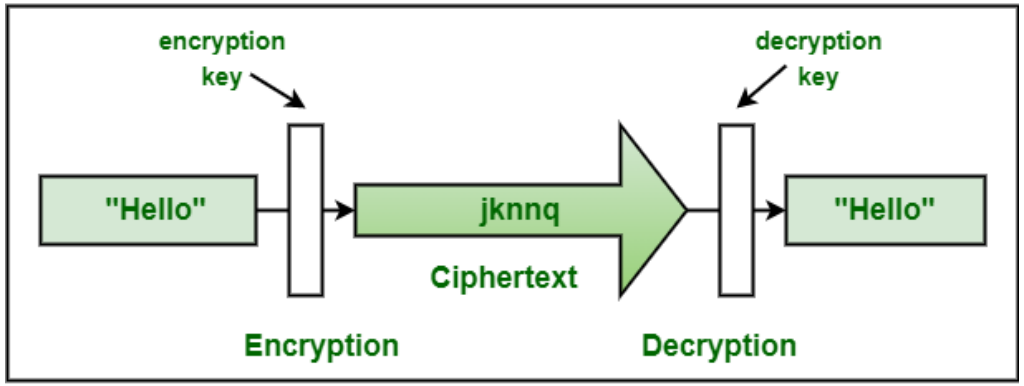
Workers of Bletchley Park circa 1938

Background: Cryptanalysis



- Will rely on user input, functions, for loops, conditional statements, ASCII codes, string methods, and comparison operators to create our own cryptography code!
- **Goal:** Write a Python program that enables us to encrypt a message that can only be read by someone with the secret key.

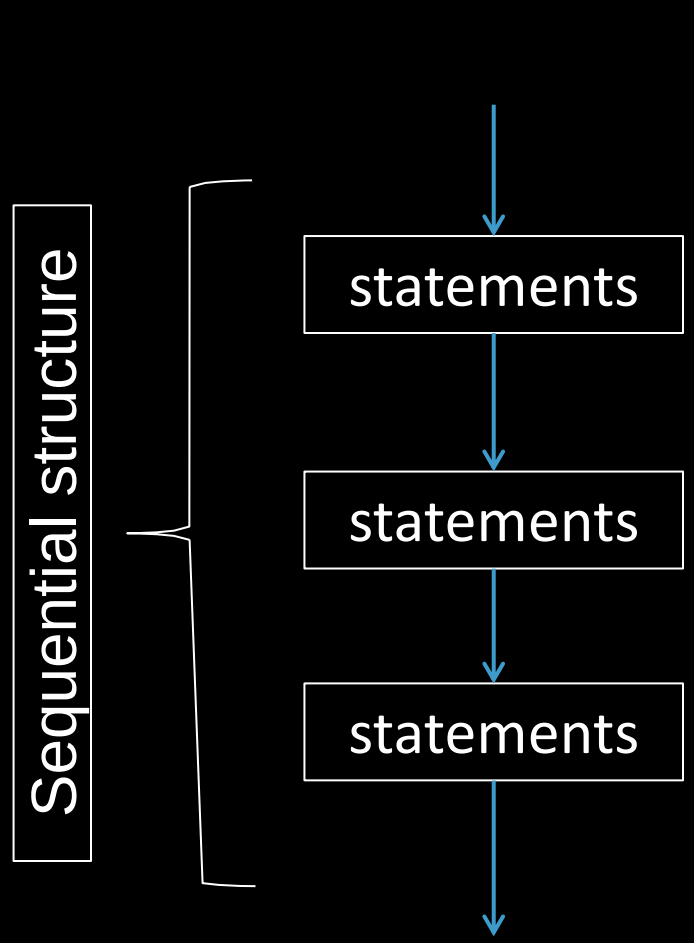
Learning Objectives

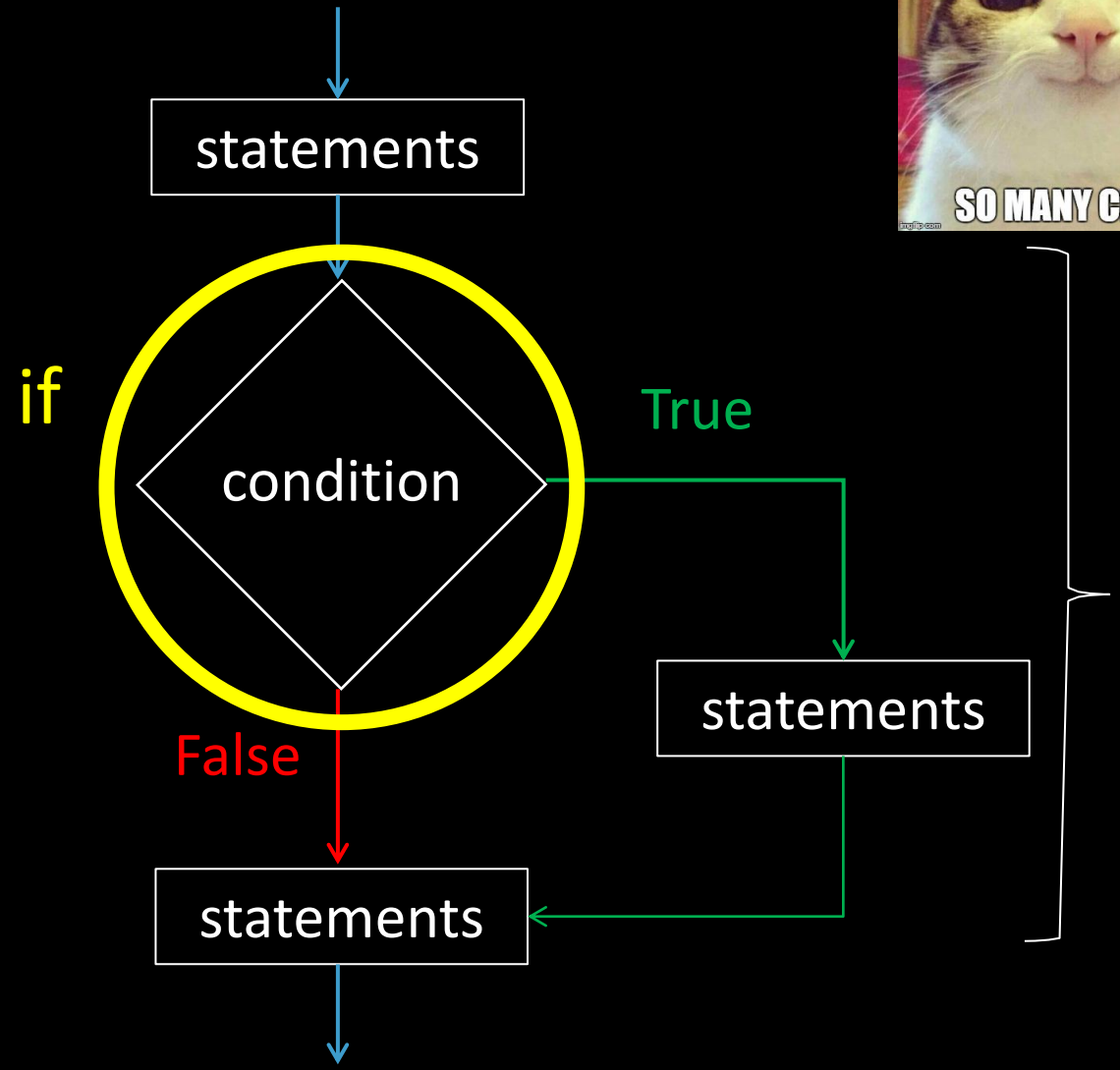


- Practice combining all topics covered so far with an out-of-box application

Some Reminders

RECAP: Making Choices





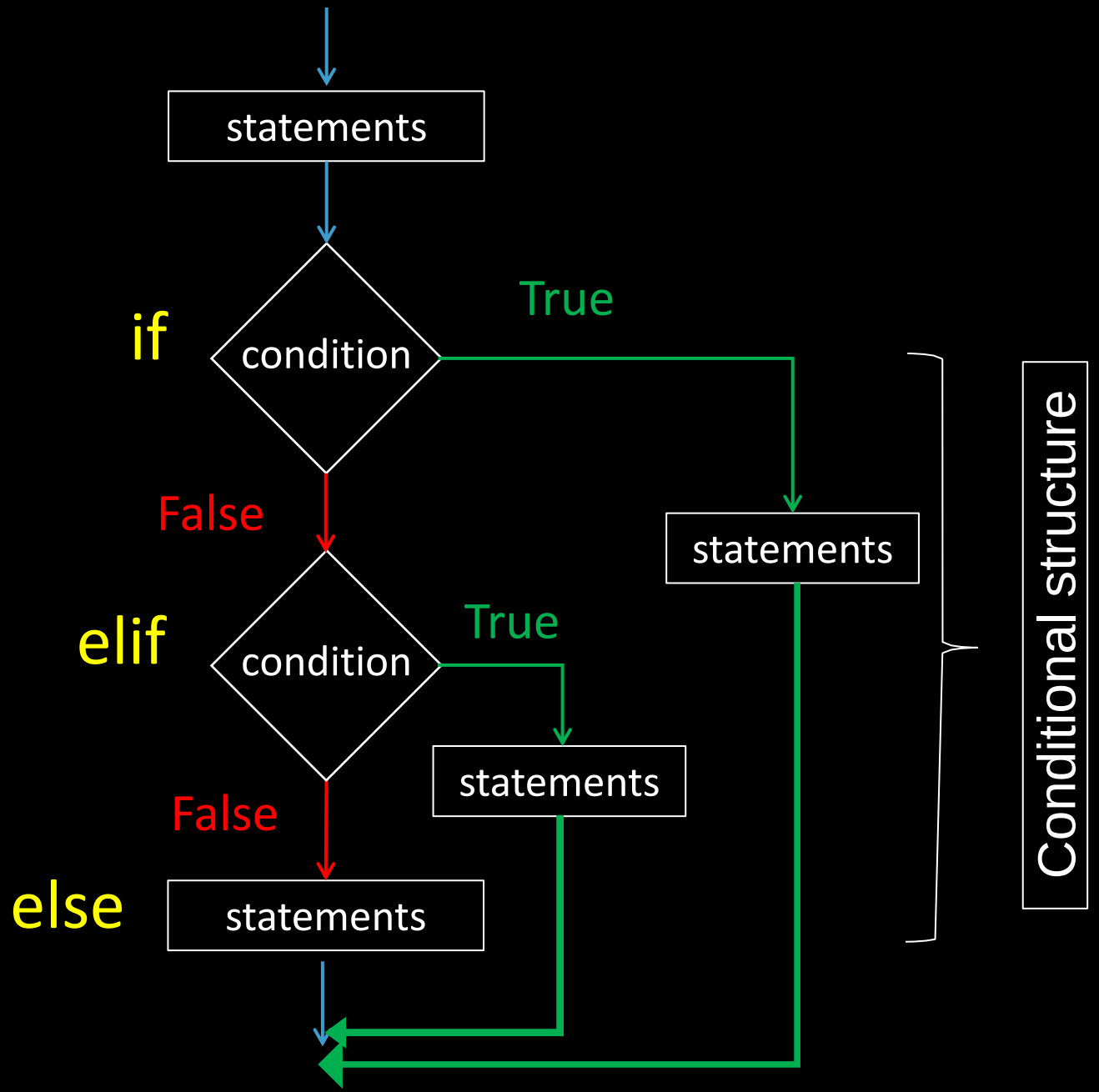
Adding the elif (else if) statement

- The most general form of the if conditional statement is:

```
if condition1:
    body1
elif condition2:
    body2
elif conditionN:
    bodyN
else:
    other_body
```

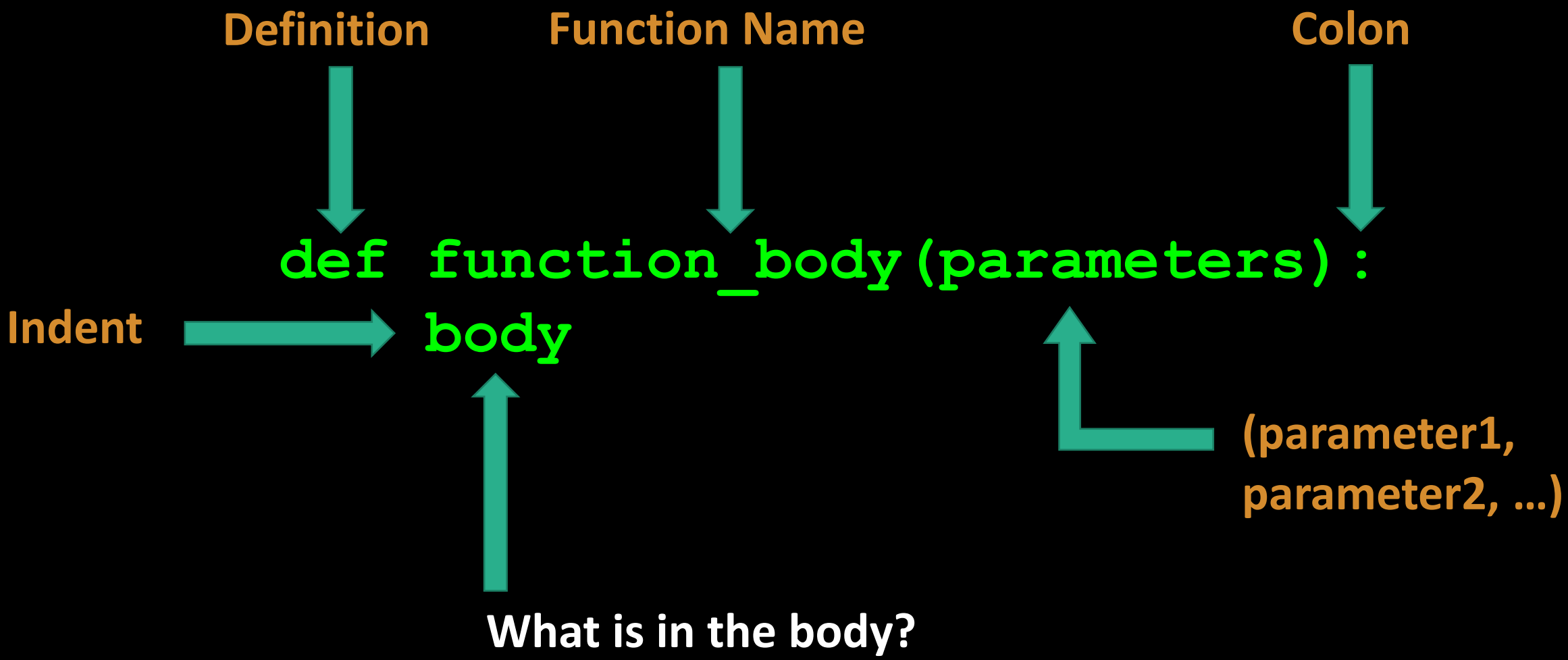
- Note the colons (:) and the indents!
- ONLY 1 body will be executed.
  - if statement is True, execute body1, exits if structure
  - if statement is False, continue to elif statement
  - elif statement is True, execute elif body, exits if structure
  - elif statement is False, continue to next elif statement
  - All if's and elif's are False, execute else statement

Making Choices



Review: Functions & User Input

Function Definitions



Function Definitions

```
def function_body(parameters):
```

1. """DOCSTRING""" (optional)

2. Code the does the thing

3. return [expression]
   The return statement is
   options and if it is not
   included, it's the same as
   writing return None

Calling Functions

- The general form of a function call:

```
function_name(arguments)
```

- Terminology
  - *argument*: a value given to a function.
  - *pass*: to provide an argument to a function.
  - *call*: ask Python to execute a function (by name).
  - *return*: give a value back to where the function was called from.

In **Python** names of variables and functions use low case and underscores.

```
function_name
Function_Name
FunctionName
```

Input

- Python has a built-in function named **input** for reading text from the user.
- The general form of a **input** function call:

```
input(argument)
```

- The argument is the text you want displayed to the user.
  - *"What is your name?"*
- The value returned by the **input** function is always a string.