Computer Science Fundamentals

Seminar 2

Lecturer: Olga Yugay

Seminar objectives

Part A

1. Positional notation, conversion between number systems
2. Binary addition

Part B

1. Practise python and conversion, variables, data types, boolean expressions
2. Python conditionals and recursion
3. Practise basic cmd commands
4. Practise basic git commands

Part A: Lecture review

Positional notation

Positional notation is a system of expressing numbers in which [1]

- the digits are arranged in succession (0, 1, 2, 3…)
- the position of each digit has a place value
- the number is equal to the sum of the products of each digit by its place value.

9	4	3
$9*10^2$	$4*10^1$	$3*10^0$
900	40	3
900+40+3=943		

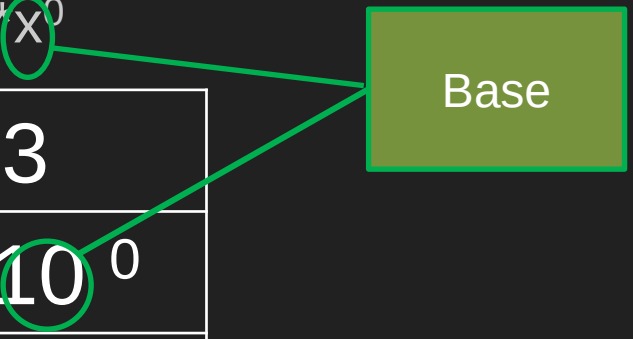
Place value

[1] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning,

Positional notation

- The rightmost digit represents its value multiplied by the base to the zeroth power, the digit to the left represents its value multiplied by the base to the first power
- Number 943 demystified: $9*x^2+4*x^1+3*x^0$

9	4	3
$9*10^2$	$4*10^1$	$3*10^0$
900	40	3
900+40+3=943		



Convert from decimal to other number systems

The algorithm:

While the quotient is not zero
Divide the decimal number by the new base
Make the remainder the next digit to the left in the answer
Replace the decimal number with the quotient [2]

[2] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 3.

Example:

Let’s try to convert 1632 in decimal to its hex equivalent

	Remainder	Hex
1632/16=102		

Example:

Let’s try to convert 1632 in decimal to its hex equivalent

	Remainder	Hex
1632/16=102	0	0
102/16=6	6	6
6/16=0	6	6

Hex: 660

Arithmetic in binary

Recall arithmetic in decimal

- Basic addition in decimal (base 10)

  0+1=1

  1+1=2

  2+1=3

  …

  9+1=10
- Because there is no symbol for 10, we reuse same digits.
- The rightmost digit reverts to 0 and there is a carry into next position to the left [4]

[4] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 3.

Recall arithmetic in decimal

- Basic addition in decimal (base 10)

  0+1=01

  1+1=02

  2+1=03

  …

  9+1=10

- Because there is no symbol for 10, we reuse same digits.
- The rightmost digit reverts to 0 and there is a carry into next position to the left [5]

[5] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 3.

Arithmetic in binary

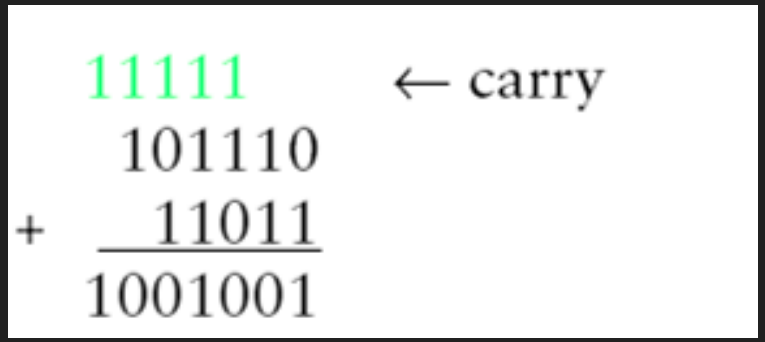
- The rules of binary arithmetic same as decimal
- But run out of digits much sooner [6]

Example 1:

0+1=1

1+1=10

Example 2:



[6] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 3.

Subtraction in binary

The subtraction rules in binary are simple, same as with decimal system:

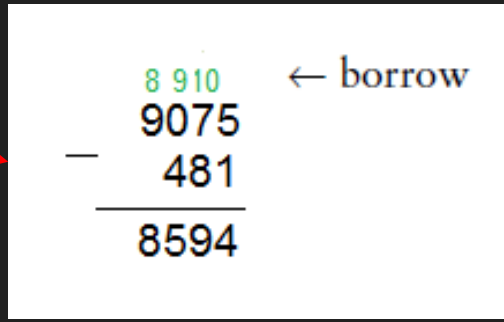
1-0=1

1-1=0

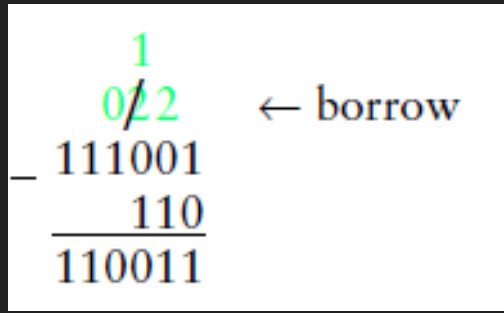
Note, when you try to subtract a larger digit from a smaller one, such as 0-1, you have to borrow one from the next left digit of the number from which you are subtracting.

More precisely, you borrow one power of the base. So in base 10, when you borrow, you borrow 10. The same logic applies to binary subtraction.
Every time you borrow in a binary subtraction, you borrow 2, which is the power of the base 2 (binary numbers).[7]

Decimal subtraction



Binary subtraction



[7] Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 3.

Tasks 1-4 Binary, octal, hex

1. Enter the first link Google Spreadsheet (Tasks 1-4)
2. Create Google Sheet and copy tasks there
3. Complete the tasks listed (Review conversion, addition and subtraction on slides 4, 5 and 6)
4. Compare answers with Google Spreadsheet (Answers)

Part B

Python [8]

Number system	Function	Prefix	Example
Binary	bin()	‘0b’	0b100
Octal	oct()	‘0o’	0o17
Hex	hex()	‘0x’	0x1C

The int() function can be used to convert numbers into a base 10 integer from any base between 2 and 36 by changing the second parameter. e.g. int(number, 8)

[8] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1

int()

int(value, base)

value	A number or a string that can be converted into an integer number
base	A number representing the number format. Default value: 10

Examples:

```
x = int("12")
y = int(“0b01010”, 2)
```

[10] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1

Hexadecimal

What does it this code do?

```
hex_num1 = "0x10"
hex_num2 = "0x10"
my_sum = int(hex_num1, 16) + int(hex_num2, 16)
print(my_sum)
print(hex(my_sum))
```

Important terms

Variables are simply places to store information and to give that information a name. As the term indicates the information stored is "variable", meaning that it can change
Example: `hex_num1 = "0x10"`

Expressions are combinations of values, variables, and operators that the Python interpreter evaluates to compute a resulting value.

Statements are units of code that have an effect, like creating a variable or displaying a value. The Python interpreter executes statements to produce the effect. (When executing a statement, the interpreter evaluates any expressions included in the statement.)

Functions are named sequences of statements that perform computations. After you define a function, you can call it any number of times to have the Python interpreter execute the statements in the function. [11]

[11] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Expression vs statement

Here are some examples of expressions: [12]

```
1. 2 + 2
2. 3 * 7
3. 1 + 2 + 3 * (8 ** 9) - sqrt(4.0)
4. min(2, 22)
5. max(3, 94)
6. round(81.5)
7. "foo"
8. "bar"
9. "foo" + "bar"
10. None
11. True
12. False
13. 2
14. 3
15. 4.0
```

All of the above can be printed or assigned to a variable.

Here are some examples of statements:

```
1. if CONDITION:
2. elif CONDITION:
3. else:
4. for VARIABLE in SEQUENCE:
5. while CONDITION:
6. try:
7. except EXCEPTION as e:
8. class MYCLASS:
9. def MYFUNCTION():
10. return SOMETHING
11. raise SOMETHING
12. with SOMETHING:
```

None of the above constructs can be assigned to a variable. They are syntactic elements that serve a purpose, but do not themselves have any intrinsic “value”. In other words, these constructs don’t “evaluate” to anything.

[12] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1

Boolean expression

A boolean expression is an expression that is either true or false. The following examples use the operator == , which compares two operands and produces True if they are equal and False otherwise: [13]

```
>>> 5 == 5
```

```
True
```

```
>>> 5 == 6
```

```
False
```

True and False are special values that belong to the type bool ; they are not strings:

[13] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Relational operators[14]

Symbol	Name	Example	Value
==	Equal to	2==2	True
!=	Not Equal to	3!=2	True
>	Greater Than	3>2	True
<	Less Than	3<2	False
>=	Greater Than or Equal to	3>=2	True
<=	Less Than or Equal to	2<=2	True

[14] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Logical operators [15]

Symbol	Name	Example (x = 5)	Value
and	Returns True if both statements are true	x <= 5 and x < 10	True
or	Returns True if one of the statements is true	x <= 5 or x >4	True
not	Reverse the result, returns False if the result is true	not(x <= 5 and x < 10)	False

Python is not very strict. Any nonzero number is treated as True

[15] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Arithmetic operators [16]

Symbol	Name	Example	Value
+	Addition	8+3	11
-	Subtraction	8-3	5
*	Multiplication	8*3	24
/	Division	8/2	4
//	Floor Division	9//2	4
**	Power	8**3	512
%	Modulus	8%3	2

More on operators https://www.w3schools.com/python/python_operators.asp

[16]Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Floor division [17]

Floor division // divides two numbers and rounds down to an integer

Example: Run time of a movie is 105 minutes. You want to know the runtime in hours.

```
>>> minutes = 105
>>> minutes / 60
1.75
```

But hours are usually not recorded with decimal points. ⇒ Try floor division

Floor division returns the integer number of hours, rounding down:

```
>>> minutes = 105
>>> hours = minutes // 60
>>> hours
1
```

To get the remainder, you could subtract off one hour in minutes:

```
>>> remainder = minutes - hours * 60
>>> remainder
45
```

[17] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Modulus [18]

Modulus operator % divides two numbers and returns the remainder.

```
>>> remainder = minutes % 60

>>> remainder

45
```

Using % you can check whether one number is divisible by another, if x % y is zero, then x is divisible by y

[18] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2.

f-string

F-strings provide a concise and convenient way to embed python expressions inside string literals for formatting. [19]

Example:

```
name = “Olga”

module = “CFS”

print(f"Hello, My name is {name} and I'm teaching {module}")
```

[19] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Getting input

input() - is built-in function to get user input. The result of it has to be assigned to the variable

int() - built-in function converts the specified value into an integer number [20]

```
year = input("What year were you born?")

year_now = 2021

print("You are", year_now-int(year), "years old")
```

[20] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Task 5: Python

Write a small Python program to convert decimal into other number systems. Take input from the user. Experiment with int(), bin(), oct(), hex().

Create a PyCharm project e.g. CSF/seminar2/

Make a comment #task5 inside main.py

Task 6

Write a Python program to find whether a given number (accept from the user) is even or odd, print out an corresponding message to the user. Make a comment #task6 inside main.py inside CSF/seminar2/ project

Hint:

```
if   condition   :
            print(“This is odd number”)
else:
            print(“This is even number”)
```

Function

In Python a function is defined using the def keyword: [21]

```
def my_function():

  print("Hello!")
```

The function can be called multiple times by using function name followed by parenthesis

```
my_function()
```

[21] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Function

Function is a named sequence of statements that performs a computation. When you define a function, you specify the name and the sequence of statements. Later, you can “call” the function by name. [22]

```
def calculateMark(weight, mark):

        result = weight*mark

        return result

calculateMark(0.4, 65)

calculateMark(0.6, 55)
```

Function definition

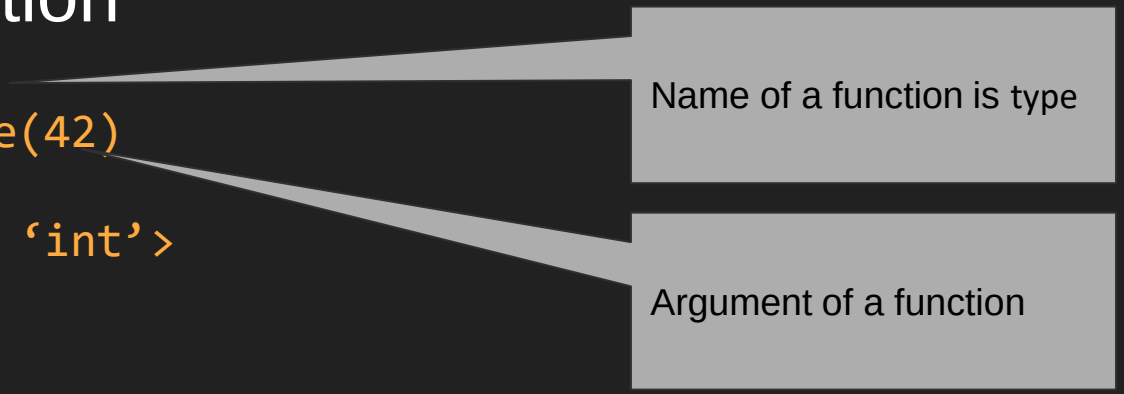
Function is called

[22] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Function

```
>>>type(42)

<class ‘int’>
```



It is common to say that a function “takes” an argument and “returns” a result. The result is also called the **return value**.

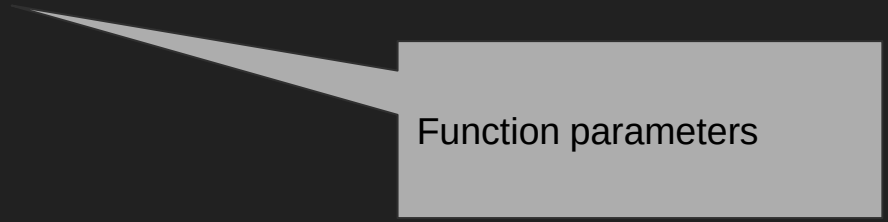
Function: interface

interface of a function is a summary of how it is used:

- What are the parameters?
- What does the function do?
- What is the return value?

```
def calculateMark(weight, mark):

        return weight*mark
```



Function: example

```
def calculateMark(weight, mark):

        return weight*mark


cwWeight = 0.4

cwMark =65

calculateMark(cwWeight, cwMark)

calculateMark(0.6, 55)
```

Parameter. A name used inside a function to refer to the value passed as an argument.

Argument. A value provided to a function when the function is called. This value is assigned to the corresponding parameter in the function.

void function vs value returning function [23]

Void function

Function that performs an action but **does not** return any value. *In fact None a special value is returned by void functions.

```
id = 000123

def printID():

        print(id)
```

Value returning function

Function that performs an action and returns a value

```
def calculateMark(weight, mark):

        return weight*mark
```

[23] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Task 7

Comment the previous tasks using Ctrl+/
Modify previous tasks 5 and 6 into functions, e.g. conversion and even_odd functions. Call the functions.

Global variables

Unlike local variables, which disappear when their function ends, global variables persist from one function call to the next. [24]

```
globalVariable = 10

def functionToCallGlobalVariable:

        print(globalVariable)

def someFunction():

        localVariable = 2
```

[24] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Global variables

Global variables are usually used for flags => boolean variables that indicate (“flag”) whether a condition is true. [25]

> For example, some programs use a flag named `verbose` to control the level of detail in the output:

```
verbose = True

def example1():

    if verbose:

    print('Running example1')
```

[25] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Global variables

To reassign a global variable inside a function you have to declare the global variable before you use it: [26]

```
been_called = False

def example2():

    global been_called

    been_called = True
```

The **global statement** tells the interpreter something like, “In this function, when I say been_called, I mean the global variable; don’t create a local one.”

[26] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Global variables

Here's an example that tries to update a global variable: [27]

```
count = 0
def example3():
    count = count + 1 # WRONG
UnboundLocalError: local variable 'count' referenced before
assignment
```

when you make an assignment to a variable in a scope, that variable becomes **local** to that scope and shadows any similarly named variable in the outer scope
=> count became local

How to fix it??

[27] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-2

Solution to previous problem

```
def example3():
    global count
    count += 1
```

Be careful

Global variables can be useful, but if you have a lot of them, and you modify them frequently, they can make programs hard to debug.

Task 8

Write additional comments for the functions that you have written.
Define which function takes an argument. Specify where the function call takes place. Identify what code is the argument and what code is the parameter.

git

Configure git

- Open git-cmd.exe
- Type the following commands

```
git config --global user.name "Your name"
git config --global user.email "youremail@gmail.com"
```

You first type "git", followed by a command – "config" and pass an option, which is "--global" in the code above.

The option "--global" means that you set your username and email for Git globally on your computer. No matter how many projects with separate local repositories you create, Git will use the same username and email to mark your commits.

Let’s try it

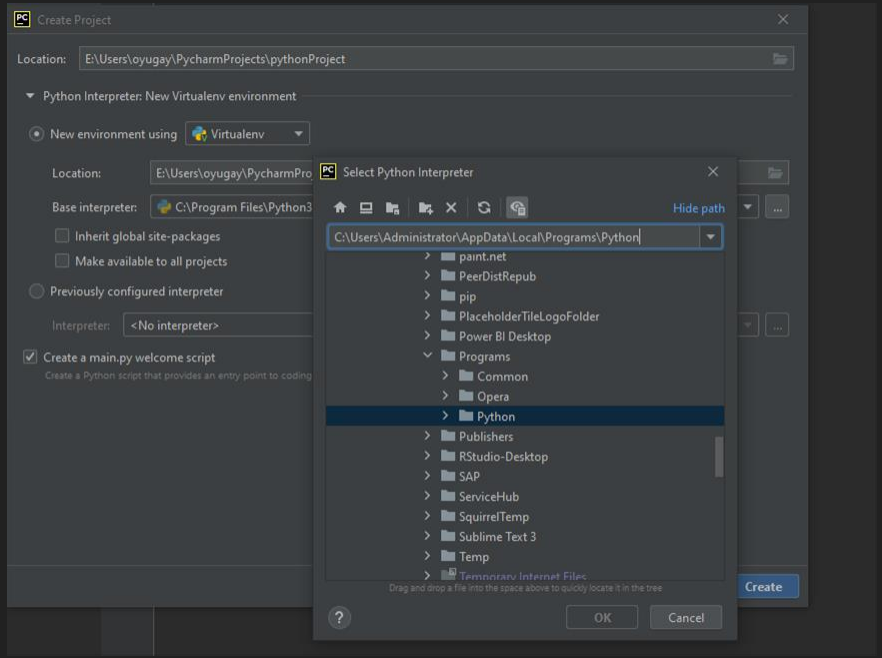
1. Go to directory

   ```
   cd PycharmProjects\CSF\
   ```

   Type `git init` - to initialize, this command creates new git repository
1. Type `git status` - to check the status of the files
2. `git add --all`
3. `git commit -m “first commit, seminar 2”` - add meaningful message

If there are problems running Pycharm

C:\Users\Administrator\AppData\Local\Programs\Python

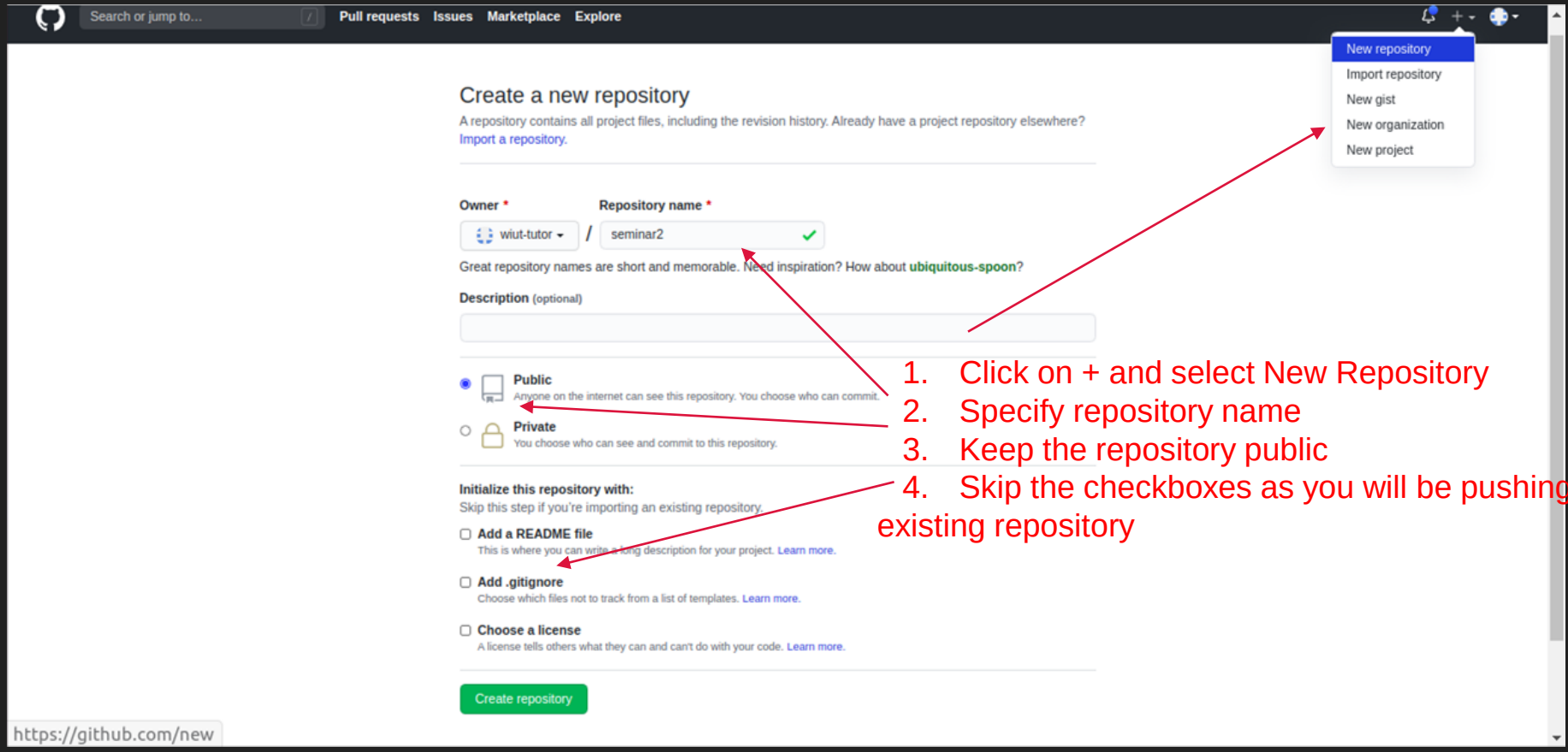


Homework 0:

Register on Github with your google account (containing ID, not your name)

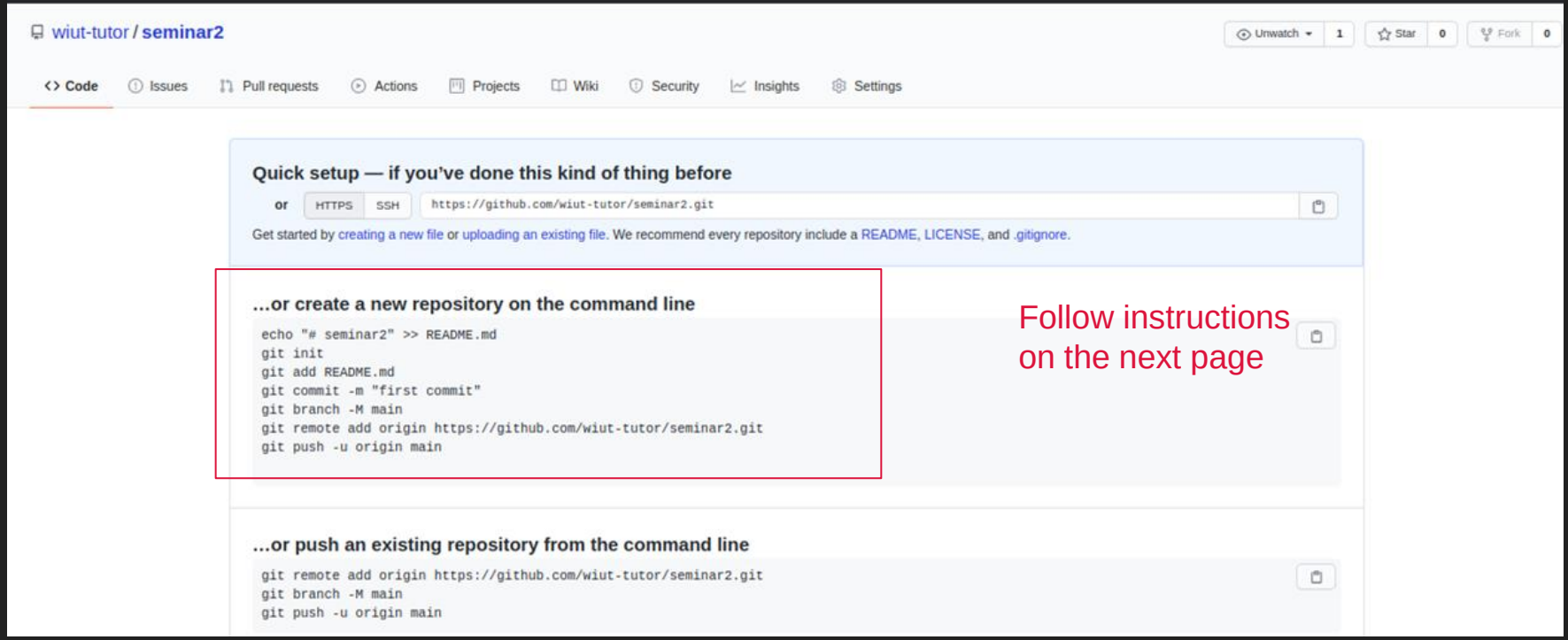
Push all you previous Tasks 5-8 tasks to your Github. Remember to register using your google profile

Github instructions 1



1. Click on + and select New Repository
2. Specify repository name
3. Keep the repository public
4. Skip the checkboxes as you will be pushing existing repository

Github instructions 2



Homework task 1

Call your function from Task 5 and 6 three times with different kinds of arguments: a value, a variable, and an expression. Identify which kind of argument is which in comments.

Commit changes and push to GitHub repository

Homework 2

Use f-string, input(), int() functions to calculate mark for the CSF module.

40% CW, 60% exam.

Push to GitHub repository

Homework 3

Complete the following task
https://docs.google.com/document/d/15TOOtQxjrDtLoHHB3H0OTDyQFcIfZ1Yjc_3s_W8P5Mo/edit?usp=sharing

Push task to GitHub repository

Recommended actions

Downey, A. (2015). *Think Python: How to think like a computer scientist.* Green Tea Press. Ch1, 2, 3, available on intranet

- Read and practice Git
  - http://rogerdudler.github.io/git-guide/
  - https://git-scm.com/about
  - https://rubygarage.org/blog/most-basic-git-commands-with-examples
  - https://ru.atlassian.com/git/tutorials

Resources

https://realpython.com/python-data-types/

https://realpython.com/python-variables/

https://www.youtube.com/watch?v=_uQrJ0TkZlc&t=301s

References

- Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 2.
- Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1
- Atlassian. 2022. Git Tutorials and Training | Atlassian Git Tutorial. [online] Available at: <https://atlassian.com/git/tutorials> [Accessed 19 April 2022].
- Git-scm.com. 2022. About - Git. [online] Available at: <https://git-scm.com/about> [Accessed 19 April 2022].
- Rogerdudler.github.io. 2022. git - the simple guide - no deep shit!. [online] Available at: <http://rogerdudler.github.io/git-guide/> [Accessed 19 April 2022].
- Rubygarage.org. 2022. Most Basic Git Commands with Examples. [online] Available at: <https://rubygarage.org/blog/most-basic-git-commands-with-examples> [Accessed 19 April 2022].
- Youtube.com. 2022. Python Tutorial - Python Full Course for Beginners. [online] Available at: <https://www.youtube.com/watch?v=_uQrJ0TkZlc&t=301s> [Accessed 19 April 2022].
- Python, R., 2022. Variables in Python – Real Python. [online] Realpython.com. Available at: <https://realpython.com/python-variables/> [Accessed 19 April 2022].