Computer Science Fundamentals

Seminar 3

Agenda

Part A

- Review how to register on GitHub and get started
- Local operations
  - Git commit
  - Git branch and merge
- Remote
  - Push

Part B

- Lecture based Q&A (e.g. Text, image and graphics, sound representation)
- Run-length encoding
- Huffman encoding
- Text representation
- Python: Conditionals and Recursion
- Python: Chained conditionals
- Python: Nested conditionals
- Python Recursion
- Python: H/w iteration and strings

Part A: Git

Register on Github and share your details

Your homework was to register on Github.

If you did not manage to setup git and GitHub this is a small video about it

Connecting to GitHub with SSH

- With SSH keys, you can connect to GitHub without supplying your username and personal access token at each visit.

Connecting to Github with SSH

Detailed instructions are here:
https://docs.github.com/en/authentication/connecting-to-github-with-ssh

In the Git Bash run the following

```
ssh-keygen -t ed25519 -C "wiut.tutor@gmail.com"

eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id_ed25519

clip < ~/.ssh/id_ed25519.pub
```

Replace with **your** gmail

Go to https://github.com/settings/keys and paste the generated key

Task 0: Review previous seminar

1. Run Git Bash and go to folder PyCharmProjects\CSF\ `cd PyCharmProjects\CSF\`
2. create seminar3 folder `mkdir seminar3`
3. Initialize git repository `git init`
4. Check the user.name (use git config user.name)

if name and email not setup run:

```
git config --global user.name "YourID"
git config --global user.email "youremail@wiut.uz"
```

1. Add a file to the folder with name `echo "seminar3" >> README.md`
2. Stage and commit the changes with message

```
git add .
git commit -m “initial commit”
git branch -M main
```

1. View history `git log`
2. Push the updates to GitHub
    a. Login to Github and create a repository
    b. Set a remote git remote add origin <your Github repository>
    c. Push updates git push -u origin main

*If you accidently specified wrong repository use git remote set-url origin <repository-name>

Alternative option

Try connecting to Github through Pycharm

And committing and pushing through Pycharm

git branch

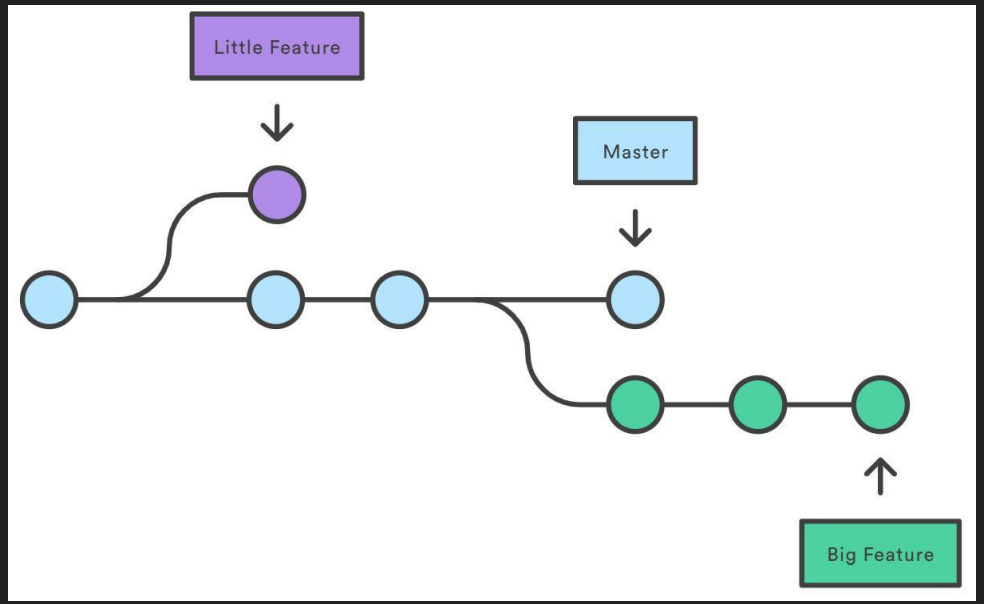
Branching is a feature available in most modern version control systems. In Git, branches are a part of your everyday development process. Git branches are effectively a pointer to a snapshot of your changes.

When you want to introduce a new feature or fix a bug—no matter how big or how small—you create a new branch to encapsulate your changes.

This makes it harder for unstable code to get merged into the main code base, and it gives you the chance to clean up your future's history before merging it into the main branch. [1]

[1] Git scm.com. 2022. About Git. [online] Available at: <https://git scm.com/about> [Accessed 19 April 2022].

General picture of branching concept [2]



[2] Git scm.com. 2022. About Git. [online] Available at: <https://git scm.com/about> [Accessed 19 April 2022].

Task 1: git branch

1. Go back to your seminar4/task5 directory
2. Create a branch git branch crazy-experiment
3. See what branch you currently on git branch
4. Select the branch you created git checkout crazy-experiment
5. Create a file with some text echo text>>file-on-branch.txt
6. Commit the update with corresponding message

   git add .

   git commit -m “update on branch crazy-experiment”

1. List the files on this branch: dir
2. Switch to main branch: git checkout main
3. List the files on master branch: dir

Note the different files being tracked on different branches

Task 2: push updates from branch

1. Check if you are on the needed branch: git checkout crazy-experiment
2. Push updates git push -u origin crazy-experiment
3. Check Github repository and note two branches with corresponding files
4. Make more updates and push them to remote repository

git merge

Merging in Git creates a special commit that has two unique parents. A commit with two parents essentially means "I want to include all the work from this parent over here and this one over here, and the set of all their parents.“ [3]

[3] Git scm.com. 2022. About Git. [online] Available at: <https://git scm.com/about> [Accessed 19 April 2022].

Task 3: git merge

1. Create a branch with name “branch-for-merge”
2. Add files to track by this branch
3. Merge master and branch-for-merge

```
git checkout main
```

```
git merge branch-for-merge
```

*If vim you can use ESC and Shift+zz.

1. Push the updates to Github repository
2. Check out Github repository

Part B: Lecture review and Python

Lecture based Q&A

1. How does pixel resolution affect the visual impact of an image?
2. Which produces better sound quality, higher sampling rates or lower sampling rates?
3. What are the techniques called that shrink the sizes of movies?

Task 4: Run-length encoding

Watch till 3rd minute Compression Crash Course Computer Science #21 [4]

a. How would the following string of characters be represented using run length encoding? What is the compression ratio?

AAAABBBCCCCCCCCDDDDhithereEEEEEEEEEEFF

a. What does *X5*A9 represent using run-length encoding?

[4] Crash Course. 2022. Compression: Crash Course Computer Science #21 - Crash Course. [online] Available at: <https://thecrashcourse.com/courses/compression-crash-course-computer-science-21/> [Accessed 19 April 2022].

Example of Run-length encoding in python

```
def encode(message):
    encoded_message = ""
    i = 0
    while (i <= len(message) - 1):
        count = 1
        char = message[i_of_external]
        j = i
        while (j < len(message) - 1):
            if (message[j] == message[j + 1]):
                count = count + 1
                j = j + 1
            else:
                break
        encoded_message = encoded_message + str(count) + ch
        i = j + 1
    return encoded_message

encoded_message = encode("ABBBBCCCCCCCCAB")
print(encoded_message)
```

Note function parameter

Note 2 while loops and if else

Questions:

1. What does indentation on the left mean?
2. What does ‘return’ mean?
3. What is len()?

Note function argument

[4] GeeksforGeeks. 2022. *Run Length Encoding in Python - GeeksforGeeks*. [online] Available at: <https://www.geeksforgeeks.org/run-length-encoding-python/> [Accessed 19 April 2022].

Task 5: Run-length encoding

Update the previous code to represent it in the following format:

*A1*B4*C8*A1*B1

Save the project to seminar 3 folder.

* NB: Commits can be run later. Put the code into relevant folder (../seminar4/ and make corresponding commits)

Task 6: Huffman encoding

- When we decode, how do we know how many bits to use for each character (since variable-length codes are used)?

- Given the following Huffman encoding table, decipher the bit strings below.

Huffman Code	Character
00	A
11	E
010	T
0110	C
0111	L
1000	S
1011	R
10010	O
10011	I
101000	N
101001	F
101010	H
101011	D

a. 10100010010101000100011101000100011

b. 10100100101000010001000010100110110

Task 7: Huffman encoding

Watch starting from 3 minute Compression Crash Course Computer Science #21

How many bits may be required for encoding the message ‘mississippi’ using Huffman encoding? [5]

Hint:

- First calculate frequency of characters
- Generate Huffman Tree
- Calculate number of bits using frequency of characters and number of bits required to represent those character

[5] GeeksforGeeks. 2022. Practice Questions on Huffman Encoding - GeeksforGeeks. [online] Available at: <https://www.geeksforgeeks.org/practice-questions-on-huffman-encoding/> [Accessed 19 April 2022].

Task 8: Text representation

A program has performed transcoding of message written in Russian with a length of 20 symbols originally recorded in UCS-2 (2 byte Unicode) into KOI8-R (8 bit KOI). By how much did the length of message get reduced?

NB:

KOI8-R - 8-bit character encoding, designed to cover Russian

UCS-2 - character encoding standard in which characters are represented by a fixed-length 16 bits (2 bytes). Covers characters from multiple languages.

Python

Python: Conditionals

Conditionals use boolean expressions, which have the value True or False, to decide between alternative code paths.

Conditionals are used within recursive functions to avoid infinite recursion and perform useful computations. [6]

[6] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-3

if

Allows to check conditions and change the behavior of the program accordingly.

```
if x > 0:

    print('x is positive')
```

condition

Indented statements are run if condition is **true.** No limit on the number of statements that can appear in the body, but there has to be at least one.

It might be useful to have a body with no statements (usually as a place keeper for code you haven't written yet). In that case, you can use the pass statement, which does nothing.

```
if x < 0:

    pass # TODO: need to handle negative values!
```

if else

A second form of the if statement is “alternative execution”, in which there are two possibilities and the condition determines which one runs.

```
if x % 2 == 0:
    print('x is even')
else:
    print('x is odd')
```

The alternatives are called branches, because they are branches in the flow of execution.

Chained conditional

More than two branches

```
if x < y:
    print('x is less than y')
elif x > y:
    print('x is greater than y')
else:
    print('x and y are equal')
```

Nested conditional

One conditional can also be nested within another.

```
if x == y:
    print('x and y are equal')
else:
    if x < y:
        print('x is less than y')
    else:
        print('x is greater than y')
```

Consider more concise versions

```
if 0 < x and x < 10:
    print('x is a positive single-digit number.')


if 0 < x < 10:
    print('x is a positive single-digit number.')
```

Recursions

A function that calls itself is recursive; the process of executing it is called recursion.

Stack diagrams can help you track and understand what a recursive function is doing as it calls itself. [7]

[7] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-3

recursive function

```
def countdown(n):
    if n <= 0:
        print('Blastoff!')
    else:
        print(n)
        countdown(n-1)
>>> countdown(3)
```

Beware of infinite recursion [8]

If a recursion never reaches a base case, it goes on making recursive calls forever, and the program never terminates. This is known as infinite recursion, and it is generally not a good idea.

```
def recurse():

    recurse()
```

[8] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-3

Keyboard input [9]

Python provides a built-in function called input that stops the program and waits for the user to type something. When the user presses Return or Enter , the program resumes and input returns what the user typed as a string.

```
>>> name = input('What is your name?\n')
What is your name?
Arthur, King of the Britons!
>>> name
'Arthur, King of the Britons!'
```

/n represent new line

[9] Downey, A. B., Elkner, J., & Meyers, C. (2015). Learning with python: How to think like a computer scientist. Green Tea Press., Chapter 1-3

Task 9

Write a program that will ask to input the module name and number of components on the module with corresponding weighting. It should also ask to enter mark for the components and show you the final mark.

Additionally you may include a feature, where user can input the desired overall mark for the module and program outputs the required marks for the components.

Save it to in seminar3/ folder

Push your updates to Github repository.

Homework 1

Copy the countdown function below (refer to Section 5.8 in Downey "Think Python" for details)

```
def countdown(n):
    if n <= 0:
        print('Blastoff!')
    else:
        print(n)
        countdown(n-1)
```

Write a new recursive function countup that expects a negative argument and counts “up” from that number. Output from running the function should look something like this:

```
>>> countup(-3)
-3
-2
-1
Blastoff!
```

Write a Python program that gets a number using keyboard input. (Remember to use input for Python 3 but raw_input for Python 2.) If the number is positive, the program should call countdown. If the number is negative, the program should call countup. Choose for yourself which function to call (countdown or countup) for input of zero.

Upload the your python file to Github and make your repository public.

Homework 2

Try to apply Huffman to the word “assessment”. Use word or draw on paper and take a photo and upload resulting files to Gitub repository

Homework 3

Iteration and strings

Review and try how to iterate over strings in the given link

Next lesson we will review detailed examples

Homework 4

Make presentation using Google slides about one of the following:

1. The Windows Club. 2022. PNG vs JPG vs GIF vs BMP vs TIF: Image file formats explained. [online] Available at: < https://www.thewindowsclub.com/png-jpg-gif-bmp-tif-image-file-formats > [Accessed 19 April 2022].
2. Joel on Software. 2022. The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!). [online] Available at: < https://www.joelonsoftware.com/2003/10/08/the-absolute-minimum-every-software-developer-absolutely-positively-must-know-about-unicode-and-character-sets-no-excuses/ > [Accessed 19 April 2022].
3. Medium. 2022. Under The Bonnet: Binary Representation of Integer. [online] Available at: < https://medium.com/@lynxluna/under-the-bonnet-binary-representation-of-integer-83b2a2283c58 > [Accessed 19 April 2022].
4. Medium. 2022. Under The Bonnet: Representing Characters and Strings.. [online] Available at: < https://medium.com/@lynxluna/under-the-bonnet-representing-characters-and-strings-ff7cec26ee3c > [Accessed 19 April 2022].

Recommended actions

1. Complete the exercises at the end of Ch 5, Downey, Think Python. Push to your Github repository
2. Dale, Computer Science Illuminated, Ch 3, end of the chapter questions and exercises
3. Files & File Systems: Crash Course Computer Science #20
4. Explore git and try yourself
    5. https://hackernoon.com/git-merge-vs-rebase-whats-the-diff-76413c117333
    6. https://www.atlassian.com/git/tutorials/using-branches
    7. https://learngitbranching.js.org/
    8. GIT branching and merging: https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging
9. Read article on SVG https://svgontheweb.com/, https://www.w3schools.com/graphics/svg_intro.asp
10. Character Encoding in Depth
11. https://medium.com/@aniboaz/animate-svg-4fa7dd00e860
12. https://www.joelonsoftware.com/2003/10/08/the-absolute-minimum-every-software-developer-absolutely-positively-must-know-about-unicode-and-character-sets-no-excuses/
13. http://kunststube.net/encoding/

References

- Atlassian. 2022. *Git Branch | Atlassian Git Tutorial*. [online] Available at: <https://www.atlassian.com/git/tutorials/using-branches> [Accessed 19 April 2022].
- Cottle, P., 2022. *Learn Git Branching*. [online] Learngitbranching.js.org. Available at: <https://learngitbranching.js.org/> [Accessed 19 April 2022].
- Crash Course. 2022. *Compression: Crash Course Computer Science #21 - Crash Course*. [online] Available at: <https://thecrashcourse.com/courses/compression-crash-course-computer-science-21/> [Accessed 19 April 2022].
- GeeksforGeeks. 2022. *Practice Questions on Huffman Encoding - GeeksforGeeks*. [online] Available at: <https://www.geeksforgeeks.org/practice-questions-on-huffman-encoding/> [Accessed 19 April 2022].
- GeeksforGeeks. 2022. *Run Length Encoding in Python - GeeksforGeeks*. [online] Available at: <https://www.geeksforgeeks.org/run-length-encoding-python/> [Accessed 19 April 2022].
- GeeksforGeeks. 2022. Iterate over characters of a string in Python - GeeksforGeeks. [online] Available at: <https://www.geeksforgeeks.org/iterate-over-characters-of-a-string-in-python/> [Accessed 21 April 2022].
- Git-scm.com. 2022. *Git - Basic Branching and Merging*. [online] Available at: <https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging> [Accessed 19 April 2022].
- Medium. 2022. *Character Encoding in Depth*. [online] Available at: <https://medium.com/tech-tajawal/https-medium-com-tech-tajawal-character-encoding-in-depth-6f1df87888d8> [Accessed 19 April 2022].
- Svgontheweb.com. 2022. A Practical Guide to SVGs on the web. [online] Available at: <https://svgontheweb.com/#animating> [Accessed 19 April 2022].

References

- Hackernoon.com. 2022. *Git Merge vs. Rebase: What's the Diff?* | *HackerNoon*. [online] Available at: <https://hackernoon.com/git-merge-vs-rebase-whats-the-diff-76413c117333> [Accessed 19 April 2022].
- The Windows Club. 2022. *PNG vs JPG vs GIF vs BMP vs TIF: Image file formats explained*. [online] Available at: <https://www.thewindowsclub.com/png-jpg-gif-bmp-tif-image-file-formats> [Accessed 19 April 2022].
- W3schools.com. 2022. *SVG Tutorial*. [online] Available at: <https://www.w3schools.com/graphics/svg_intro.asp> [Accessed 19 April 2022].
- Joel on Software. 2022. *The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)*. [online] Available at: <https://www.joelonsoftware.com/2003/10/08/the-absolute-minimum-every-software-developer-absolutely-positively-must-know-about-unicode-and-character-sets-no-excuses/> [Accessed 19 April 2022].
- Medium. 2022. *Under The Bonnet: Binary Representation of Integer*. [online] Available at: <https://medium.com/@lynxluna/under-the-bonnet-binary-representation-of-integer-83b2a2283c58> [Accessed 19 April 2022].
- Medium. 2022. *Under The Bonnet: Representing Characters and Strings.*. [online] Available at: <https://medium.com/@lynxluna/under-the-bonnet-representing-characters-and-strings-ff7cec26ee3c> [Accessed 19 April 2022].