COMP 1510 Assignment #2: Computational Thinking!

Christopher Thompson
`chris_thompson@bcit.ca`

Due **Friday January 28th at or before 23:59:59**

1 Introduction

The first few weeks of CST can be a bit of a blur. There's a lot to process: a new program, new classmates, seven courses (well, six, I think COMP 1100 is pretty easy), and the famous BCIT workload. And we're doing it partially online, of course. In 1510, we took a deep breath and dove in. We've begun our exploration of the fundamentals of programming using Python, a very popular multi-paradigm high-level language.

Computational thinking forms the underpinning of good programming. We need to decompose large problems into smaller, more easily solved problems. We identify the data we will be manipulating and their types, and we seek patterns in the processes and solutions we use. We always try to generalize our solution through the use of abstraction. And automating as much as possible with clear, correct algorithms is a critical element of computational thinking too.

For your second assignment, I challenge you to design flowcharts that solve the following problems. Use the official flowchart symbols. Use the four fundamental control structures. Generate a solution to each problem. I seek solutions that are simple and concise.

2 Submission Requirements

1. **This take home assignment is due no later than Friday January 28th at or before 23:59:59.**
2. Late assignments will not be accepted for any reason.
3. This is an individual assignment. I strongly encourage you to share ideas and concepts, but sharing charts or submitting someone else's work is not allowed.
4. Commit and push each solution after you finish it. There should be 4 commits at minimum: the first commit contains your edited README.md information, and one commit for each PDF file you add to the project.
5. You must also tell me how you tried to use the four elements of computational thinking (decomposition, pattern matching, abstraction, automation) to construct a solution that is elegant. An elegant solution is a solution that minimizes complexity and maximizes clarity. Use sequence, selection, repetition, and indirection.
6. **For each problem, there is a labeled section of the README.md where you can tell me how you applied any/all of the four elements of computational thinking to the problems you chose**.

3 Project Setup

Here is how to get started:

1. **I have created a new project template in GitHub Classroom**. click this link which will take you to the project template in GitHub Classroom. You will need to be logged in to GitHub in order to do this:

   `https://classroom.github.com/a/1sKjHOka`

2. GitHub Classroom will clone the project and create a new code repository in your GitHub account. Go ahead and visit it by clicking the URL.
3. Do you see the button at the top of the repository named Code? We're going to use that to get the URL of your new GitHub repo so we can clone it to your computer. Click Code. In the dropdown window that opens, copy the URL.
4. Open PyCharm. When PyCharm's welcome window appears, select Get from VCS and paste the URL from GitHub here. Ensure the destination is NOT nested inside an existing project.
5. This will copy the GitHub repository from your GitHub account to your computer. The git version control maintains a link between the version on your computer and the version on GitHub. I have access to the version on GitHub via GitHub Classroom and that's the version I will mark.
6. **The first thing I want you to do is enter your personal information into the README.md file**. It is a markdown file. You can apply formatting to the file to improve its organization and readability. Save your change(s). Notice that the file changes colour to blue in the list of files in PyCharm? That means there is a change that has not yet been saved into version control (git).
7. We saved the change to the file, but we still need to take a snapshot of the change you made to the file and store the snapshot in version control. We call this committing the change. Then we need to push the change to your repository in GitHub on the cloud. PyCharm makes this really easy!
8. In the PyCharm main menu, select Git and then Commit. Alternatively, choose the green checkmark in the upper right corner of the PyCharm. When we commit, git will take a snapshot of the change you made to the README.md file.
9. The dialog box that opens wants you to confirm the change. **Enter a comment, but don't press Commit yet! Your comment should begin with a verb that tells us what the code change does. In this case, "Add personal information to README.md" is good.**
10. When we use version control we say that we commit changes. That means we ask git to take a snapshot, i.e., we commit the current state of the project to memory. But we have to push the change(s) to the cloud, too. We do this by very carefully selecting the dropdown menu beside the Commit button and choosing Commit and Push. This will push your change to GitHub so I can see it.

4 Style Requirements

1. **Each flowchart must be in its own PDF file** which is named after the problem (the names are provided). You must drag and drop each completed PDF file into the correct folder in your PyCharm project, add it to git version control, commit the addition, and then push from PyCharm to the repo in GitHub so I can pull your solutions and grade them.
2. The flowcharts must not be hand-written. Free online options include Lucidchart, draw.io, and Visme (there are lots of others, too!).
3. Your flowcharts may only use the symbols we discuss in class, in lab, and on Slack:
   (a) Beginning and ending termini (pills)
   (b) Rectangle for process
   (c) Rectangle with doubles vertical sides to denote a call to a helper flowchart if you use indirection
   (d) Diamond to represent a decision
   (e) Parallelogram for printing to the screen or acquiring user input from a keyboard

(f) Arrows to indicate movement from one step to another

4. Consider using colour and/or font and/or size and/or other visual channels if necessary to convey additional meaning to the process you are mapping. Ensure identifiers are meaningful and easy to understand. Keep things simple.

5 These are fun!

Select 2 of the 3 following problems, and for each of the 2 problems you select, create a flowchart or collection of flowcharts. Each solution must be submitted as a 1-page PDF. The PDF can be any size.

1. **Lucas.pdf:** The Lucas sequence is a famous sequence of numbers. The first value in the sequence is 2. The second value in the sequence is 1. The third value, and every value afterward, is the sum of the previous 2 values. It grows quite quickly: 2, 1, 3, 4, 7, 11, 18, 29, 47, 76, 123, and so on. Ask me for a positive integer, n. Don't let me give you anything else – keep asking me for a positive integer until I give you a positive integer. Then calculate and tell me the nth Lucas number.

2. **Conversion.pdf:** I would like to convert a positive integer n from base 10 to some other destination base. The destination base must be between base 2 and 9 inclusive (we write [2, 9] to denote this). You must ask me for a base 10 integer. If I give you a negative base 10 integer, tell me I'm not allowed and then make me choose again. Don't let me give you anything but a positive base 10 integer. I can give you any positive base 10 integer. There are no limits. Then you must ask me what the destination base will be. For the destination base, don't accept anything other than an integer in the range [2, 9], keep asking me until I give you what you need. Then you must tell me what the result is. Show all your steps.

3. **Sort.pdf:** I have a hand of playing cards. These cards are all the same suit. There are 10 cards in my hand, randomly selected and shuffled. There may be duplicates. I want to sort them. I want to sort them so the smallest card is all the way on the left, and the largest card is all the way on the right.We will use this sorting order for cards: A < 2 < 3 < 4 < 5 < 6 < 7 < 8 < 9 < 10 < J < Q < K.

6 These are challenging!

Select 1 of the 2 following problems, and create a flowchart or collection of flowcharts. The solution must be submitted as a 1-page PDF. The PDF can be any size.

1. **Crane.pdf:** Teach me how to fold an origami crane. I will start with a single piece of gorgeous square origami paper with a lovely pattern on the 'good' side.

2. **Breakfast.pdf:** Tell me how to make fried eggs and toast. I like my eggs sunny side up and I like my toast burnt.

7 Grading

Your first assignment will be marked out of 10. For full marks, you must:

1. **(1 mark)** Correctly set up this assignment and use git and GitHub to commit and push 3 PDF documents to your solution repository in the manner described above.

2. **(6 marks)** Correctly represent the logic for each problem in a flowchart submitted as a PDF (2 marks per problem).

3. **(3 marks)** Describe the elements of computational thinking you applied when implementing each solution in the README.md. (1 mark per problem).

Consult the Learning Hub (D2L) and look under Assignments to see the rubric I will use when marking your assignment.

The highest grades will be reserved for submissions that minimize complexity and maximize clarity.

The very best submission wins a small LEGO set.

Please remember that this is an individual assignment. I strongly encourage you to share ideas and concepts (yes please do this!), but sharing code or screen views or submitting someone else's work is not allowed.

Good luck, and have fun!