Домашнє завдання №2

2. Визначення функціональних та нефункціональних вимог до програми, яка буде розроблятися.
Вивчення засобів стандартної бібліотеки Python для роботи з даними різних типів.
Опис даних з якими буде працювати програма.

	Завдання	Результат виконання завдання
2.1	Визначення функціональних та нефункціональних вимог до програми, яка буде розроблятися.	2 сторінки А4 12 шрифт
2.2	Описати дані, які буде отримувати програма (html, json, csv, xml...) та з якими програма буде працювати. Вказати формат, структуру згідно формату та відповідні до структури всі елементи.	2 сторінки А4 12 шрифт
2.3	Ознайомитися з документацією стандартної бібліотеки Python для роботи з даним в різних форматах 19.2. json — JSON encoder and decoder (https://docs.python.org/3.5/library/json.html) 20.2. html.parser — Simple HTML and XHTML parser (https://docs.python.org/3/library/html.parser.html) 20.4. XML Processing Modules (https://docs.python.org/3.5/library/xml.html) 14.1. csv — CSV File Reading and Writing (https://docs.python.org/3.6/library/csv.html)	
2.4	Ознайомитися з документацією інших бібліотек Python для роботи з даним в різних форматах Beautiful Soup для парсингу html (https://www.crummy.com/software/BeautifulSoup/) lxml для парсингу xml та html (http://lxml.de/)	
2.5	Описати можливості модулів, пакунків модулів, бібліотек, які будуть використовуватися для роботи з даними у програмі, що проектується.	2 сторінки А4 12 шрифт
2.6	Розробити приклад використання відповідних модулів, пакунків модулів, бібліотек для роботи з даними, які будуть отримані з мережі Інтернет.	Модуль Python
2.7	Вивчити можливості системи контролью версій Git та сховища GitHub по забезпеченню колективної робити над проектом. Описати сценарій роботи колективу розробників над проектом.	2 сторінки А4 12 шрифт
2.8	Належним чином почати оформляти файл README.md репозиторію проекта.	файл README.md

Короткі нотатки до домашнього завдання №2

2.1 Поняття функціональних та нефункціональних вимог та способи їх отримання описані в розділі3 підручника, який додано.

2.2 Потрібно навести детальний опис тих даних з якими буде в майбутньому працювати програма. Потрібно описати в якому вигляді дані будуть отримані з Інтернету - формат даних, сруктура згідно формату та всі елементи структури.

2.3, 2.4 Кваліфікований програміст повинен бути знайомим та вміти працювати з такими відомими форматами даних як json, xml, csv та html.

2.5 На основі опису формату даних буде зрозуміло, які модулі, пакунки модулів, бібіотеки доцільно використовувати для роботи з цими даними. Потрібно описати, які модулі, пакунки модулів, бібліотеки будуть використовуватися та яким чином (функції, класи, методи тощо).

2.7. Цей пункт виконують тільки ті студенти, які виконують цикл домашніх завдань в команді. В описі потрібно вказати який з варіантів організації колективної роботи обрано та описати яким чином в проекті буде організовано процес внесення змін, редагування, розподілу завдань тощо.
Для вивчення основ по колективні роботі можна скористатися, як офіційною документацією Git так й іншими матеріалами:
https://git-scm.com/book/uk/v2
http://www-cs-students.stanford.edu/~blynn/gitmagic/intl/uk/index.html

На GitHub можливі наступні варанти організації спільної роботи над проектом:

1. «Fork + Pull» . Кожен користувач GitHub може клонувати (fork) репозитарій, отримувати оновлення у цей репозитарій з оригінального репозитарію та вносити зміни у цей fork репозитарій без необхідності мати доступ до оригінального репозитарію. Зміни зроблені у fork репозитарію можуть бути додані до оригінального репозитарію його власником. Такий варіант організації спільної роботи характерний для open source проектів, оскільки довільна кількість людей може працювати незалежно і без витрат на координацію їх роботи.
2. «Спільний репозитарій». Кожен з команди розробників має доступ та права на внесення змін у репозитарій. Для розмежування версій та змін використовуються гілки (branches). Такий варіант організації спільної роботи характерний для невеликих команд.

2.8 Опис вмісту файла README.md та приклади його оформлення можна знайти за наступними посиланнями:

https://guides.github.com/features/wikis/
https://dbader.org/blog/write-a-great-readme-for-your-github-project
https://www.pluralsight.com/blog/software-development/github-tutorial

Файл README.md повинен бути в кожному з репозиторіїв та повинен містити потрібну для користувача інформацію. Для початку оформлення файлу потрібно сформувати структуру файла, вказати назву проекту та навести його короткий опис. Зрозуміло, що для більшості розділів файла інформація поки що відсутня і в подальшому файл README.md потрібно буде доповнювати та за потреби редагувати. Мова файлу README.md — англійська.