SOFT2101 Principles of Software Engineering
Fall 2025
HW #4 (50 points), In groups of maximum two students
Due Date: 02nd January Friday, 23:59

IŞIK UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING

1. Work breakdown structure (WBS) of some hypothetical communications platform development project is given below in task list form:

TASK	Duration	Predecessor (Dependency)
1. Business Analysis and High Level Modeling		
1.1. Use Case Analysis and Requirements	6 weeks	
1.2. Modeling and Architectural Design	12 weeks	1.1
1.3. Database Design	4 weeks	1.2
1.4. Procurement of Project Hardware	1 week	1.3
1.5. Definition of Test Cases	5 weeks	1.1
1.6. Delivery of Project Hardware	10 weeks	1.4
2. Design and Development of the Communication Server Software		
2.1. Detailed Architectural Design	8 weeks	1.2
2.2. Development of the Switching Software	12 weeks	2.1
2.3. Component Tests	4 weeks	2.2
3. Interactive Voice Response System and Call Center Server Design and Development		
3.1. Voice Response System Framework	8 weeks	1.3
3.2. Call Center Server Software	8 weeks	3.1
3.3. Component Tests	4 weeks	3.2
4. Design and Development of Customer Relationship Management (CRM) Applications		
4.1. Visual Designs (wireframes, mock-ups)	8 weeks	1.1
4.2. CRM Frontend Development	8 weeks	4.1
4.3. CRM Backend Development	6 weeks	1.3
4.4. Component Tests	4 weeks	4.3
5. Integration Tests, Operation & Maintenance		
5.1. Communications Server Integration Tests	4 weeks	1.6 & 2.3
5.2. Corrections and Regression Tests	2 weeks	5.1
5.3. System Integration Tests	6 weeks	1.6 & 3.2 & 4.4
5.4. Corrections and Regression Tests	4 weeks	5.3
5.5. Customer Training	2 weeks	5.4
5.6. Acceptance Tests and Demonstrations	4 weeks	5.5

Using the above WBS,

- Create the Gantt chart for the project.
- How long will it take to complete this project? You may ignore inconsistencies in estimations.
- Create the Activity Network/Chart for the project, and give the critical activity path for the project.
- Assuming that every individual task is to be carried out by only one person (i.e., it is not possible to divide any of the above listed tasks among more than one person), but at the

same time every person in the team is multi-functional, roughly how many people are required so that the project can be completed with the minimum possible timeline?

- Assume that a serious, unanticipated setback occurs, and instead of taking 10 weeks, task T1.6 takes 40 weeks. What would be the effect of this setback on the overall project? What kind of contingency plan can you propose to alleviate the negative effects of such a risk?

Please use ***Microsoft Excel*** or ***Libre Office Calc*** (free counterpart of MS Excel) to create your Gantt Chart by proper labeling/coloring, and submit a ".pdf" export of your document.

2. Before starting this question, create a profile for yourself on https://github.com if you don't have one yet (Every student, i.e., every assignment group member should have his/her own profile under GitHub).

Go to your Github profile, and create a new repository (Select the Repositories tab, and click **New**). Give your repository a logical name, a brief description, and select **public**. Next, navigate on GitHub to your repository, and add your assignment partner as a collaborator for your repository (**Settings → Collaborators and Teams → Add People).**

Create a directory named **hwk4**, and a file named **homework4.txt** within **hwk4**, under your repository's root, via your repository's GitHub web page.

If your computer does not have git installed, install git from https://git-scm.com/downloads . You can accept the defaults during the installation. All work must be done through a terminal window. You may not use graphical clients for this assignment, including any of your IDE (e.g. Intellij, Eclipse, or vscode), unless the step allows it explicitly. If you have a Mac or you use Linux, open a terminal shell. If you use Windows, you can use the Command Prompt, Powershell, or open **git bash**.

At different points throughout this question, you will be asked to enter specific text strings. Please pay careful attention to these instructions as failure to follow the instructions will very likely mean your work will be marked as wrong. You will also be asked questions throughout the steps (marked **QUESTION**: in a step's text). Please put your answers in a file called **homework4.txt** that must be in the **hmk4** directory. Please label each answer with a STEP X label where X is the step letter in which the question was asked. Often, the best answer will simply be to cut and paste the information from the terminal. Please include the git command along with the results. **homework1.txt** should be a simple text document.

Do the following tasks in order:

a. Configure git with your name, email, and favorite editor. Verify git knows your name, email, and editor by having git list your configuration.

   QUESTION: What are your git config commands?

b. Clone your repository to your machine using git's `clone` command. Show the cloned repository by listing the contents of the parent directory and then creating a listing that includes the `.git` directory.

   QUESTION: What files or directories are in your local `.git` directory?

c. Edit your repository's `README.md` to describe your repository. The `README.md` is for people who are looking for a repository, not yourself. Be creative and try something beyond *this is the repository for <your_name_and_last_name>.*

   See https://guides.github.com/features/mastering-markdown/ for formatting commands.

   QUESTION: Using markdown, how would you put a clickable link to your `README.md` to the web page of The Department of Software Engineering of Işık University (https://www.isikun.edu.tr/akademik/muhendislik-fakultesi/bolumler-ve-programlar/bilgisayar-muhendisligi/programlar/lisans-programi/yazilim-muhendisligi).

d. Decide which files you do not want git to keep track of or allow to be committed to source control, and set up your `.gitignore` appropriately. You may want to use the templates others have contributed as a guide (see https://github.com/github/gitignore for a bunch of suggestions across many technologies. Java Technologies would be fine for this assignment). Commit these rules in the project. Ensure the text "setting up gitignore" appears in your commit message.

   QUESTION: What rules are in your `.gitignore`?

e. Create a branch called **feature-1**. Switch to that branch and move into the **hmk1** directory. All your work will be done in this directory and on this branch.

f. Using an IDE of your choice, write a java program that takes as input two numbers (read from the console) and prints out all the numbers that sit between these two numbers, not including the two inputs in the range.

   Once the code works, **`add`** the project files to git and **commit** them. Please ensure the text "satisfying requirement one" is in the commit message. You should not commit any local and intermediate files (e.g., Java "**.class**" files).

SOFT2101 Principles of Software Engineering
Fall 2025
HW #4 (50 points), In groups of maximum two students
Due Date: 02nd January Friday, 23:59

IŞIK UNIVERSITY
COMPUTER
SCIENCE AND
ENGINEERING

QUESTION: What are all the commits you've made on the **feature-1** branch? What files (and sub-directories, if any) are listed in your **hmk1** directory on the **feature-1** branch. What files (and sub-directories if any) are listed in your **hmk1** directory on the **main** branch.

g. Now merge **feature-1** into **main**.
h. Create a new branch called **feature-2** off **main**.
i. Create a new branch called **feature-3** off **main**.
j. Switch to the **feature-2** branch. Change the method that prints the numbers in the range so it now prints only the odd numbers in the range. Do **not** add a new method. You **must** change the existing method. Commit the changes on **feature-2**. Please ensure the text "satisfying requirement change two" is in the commit message.

QUESTION: What are all the commits you've made on the **feature-2** branch?

k. Switch to the **feature-3** branch. Change the method that prints the numbers in the range so it now prints only the even numbers in the range. Do **not** add a method. You **must** change the existing method like you just did in the above step. Commit the changes on **feature-3**. Please ensure the text "satisfying requirement change three" is in the commit message.
l. Merge **feature-2** into **main**. Resolve any merge conflicts so that **main** reflects only the functionality you created in **feature-2**.
m. Merge **feature-3** into **main**. Resolve any merge conflicts so **main** reflects the functionality you created in **feature-3**.

QUESTION: What are the differences between **main** and the other three branches? *Hint: use* ***git diff.***

n. Push all four branches (**master, feature-1, feature-2, feature-3**) to **origin**. You can verify the results by going to the web page of your repository on GitHub, and looking at the **hmk1** directory. You should be able to see all the branches and the commit log.
o. Ensure the answers to all the questions are in **homework4.txt** and push it to origin. **homework4.txt MUST** be in your **hmk4** directory and **MUST** be found on the **main** branch.
p. Write down the link to your GitHub repository you created and used throughout this assignment, into your submitted pdf document (please see below).

https://github.com/…………

SOFT2101 Principles of Software Engineering
Fall 2025
HW #4 (50 points), In groups of maximum two students
Due Date: 02nd January Friday, 23:59

Please submit your deliverable for the first question of this assignment as a ".pdf" file (the GitHub link for the answers of the second question should also be put into this file) via the Blackboard system, under the following file naming scheme:

<team_member1_name_last_name_number_team_member2_name_last_name_soft2101_hw4.zip>