**EXERCISE FOR GIT**

**Exercise 1: Git Time Capsule**

**What it is:**

The Git Time Capsule exercise is a fun activity that introduces you to the basics of Git version control. You'll create a virtual "time capsule" with a personal message or challenge for your future self.

**What you get after performing this:**

You'll learn how to initialize a Git repository, make commits, and view your commit history. Plus, you'll have a future surprise when you open your time capsule!

**How to do it this way:**

1. Create a directory called "MyTimeCapsule."

2. Initialize a Git repository: `git init`.

3. Add a text file with your message: `git add filename`.

4. Commit your message: `git commit -m "Your commit message"`.

5. Wait for a chosen time period, then revisit your time capsule with `git log` to see your message.

**Exercise 2: Collaborative Storytelling with Branches**

**What it is:**

In this exercise, you and your teammates will use Git branches to collaboratively create a story. Each participant adds a sentence to the story in their branch.

**What you get after performing this:**

You'll practice branching, committing, merging, and conflict resolution. You'll also create a unique collaborative story.

**How to do it this way:**

1. Start with a sentence or a story snippet.

2. Create a branch with your name: `git branch YourName`.

3. Add a sentence to the story and commit: `git commit -m "Your message"`.

4. Merge your branch into the main story branch: `git merge YourName`.

**Exercise 3: The Great Git Race**

**What it is:**

Get ready for some Git competition! Teams race to complete Git tasks and earn points for each completed task. The fastest team wins!

**What you get after performing this:**

You'll sharpen your Git skills, including initializing repositories, making commits, creating branches, merging, and resolving conflicts, all while having a blast in a competitive setting.

**How to do it this way:**

1. Form teams and assign a starting repository to each.

2. Race to complete tasks, like making commits or merging branches.

3. Earn points for each task completed accurately and quickly.

4. Bonus points for resolving conflicts correctly if they occur.

**Exercise 4: Git Puzzle Hunt**

**What it is:**

It's a treasure hunt with a Git twist! Solve Git-related puzzles and clues to uncover a secret message or prize.

**What you get after performing this:**

You'll enhance your problem-solving skills, critical thinking, and Git command knowledge. Plus, you'll enjoy the thrill of a puzzle adventure.

**How to do it this way:**

1. Prepare a series of Git-related puzzles and clues.

2. Start with a Git repository containing the first puzzle.

3. Solve each puzzle to reveal the next clue or Git command.

4. The final puzzle leads to a secret message or prize.

**Exercise 5: Git Code Art**

**What it is:**

Unleash your creativity with Git! Collaboratively create a piece of artwork using text characters in a Git repository.

**What you get after performing this:**

You'll practice teamwork, branching, merging, and version control while expressing your creativity through code art.

**How to do it this way:**

1. Create a Git repository for code art.

2. Each participant creates a branch and crafts a piece of code art.

3. Commit and push your code art to your branch.

4. Merge all branches into the main branch.

5. Display your final code art on a shared screen or print it out for a visual masterpiece.