Sample language - 4 Sprints Pacman

Hufflepuff

June 17, 2024

1 SPRINT 1

```
// SPRINT 1
func sprint1Tasks -> List:Task {
  params {}
  return {
    List:Task [
      Task {
        title: "Development environment setup",
        description: "As a developer, I want to have all necessary tools like compilers, development
↪  environments, and version control systems installed.",
        state: "DONE",
        members: List:Member [Member {name: "Axel", role: "Developer"}],
        tag: "Backend",
        subTasks: List:Task []
      },
      Task {
        title: "Initial maze design",
        description: "As a user, I want to create a basic maze design using graphic design tools.",
        state: "DONE",
        members: List:Member [Member {name: "Diego", role: "Designer"}],
        tag: "Frontend",
        subTasks: List:Task []
      }
    ]
  }
}

// Create task for Pacman movement
func taskMovementPacman -> Task {
    params {
        titleTask: StringParagraph
    }
    return {
        Task {
```

```
        title: titleTask,
              description: "As a developer, I want to implement the basic movement of Pacman in the
↪  maze.",
              state: "DONE",
              members: List:Member [Member {name: "Luiggy", role: "Developer"}],
              tag: "Backend",
              subTasks: List:Task []
        }
    }
}

// Print task titles
func printTasksTitle -> String {
  params { task: Task }
  return {
    task.title
  }
}

// Check task status
func taskIsDone -> Bool {
  params { myTask: Task }
  return {
    if (myTask.state == "DONE") then True
    else False
  }
}

// Function execution
do {
  taskIsDone(taskMovementPacman("Basic Pacman movement"))
  // Since do can only handle one function at a time, another example of execution is:
  map(sprint1Tasks, printTasksTitle)
}
```

2 SPRINT 2

```
// SPRINT 2
func sprint2Tasks -> List:Task {
  params {}
  return {
    List:Task [
      Task {
        title: "Spike: Research ghost movement patterns",
```

```
        description: "As a user, I want to define the movement patterns for each ghost based on studies of previous Pac-Man versions and modern adaptations. Explore AI algorithms to improve movement autonomy.",
        state: "DONE",
        members: List:Member [Member {name: "Santiago", role: "Developer"}],
        tag: "Spike",
        subTasks: List:Task [
          Task {
            title: "Experimenting with AI algorithms",
            description: "As a user, I want to implement AI algorithm prototypes to simulate autonomous and adaptive ghost movements.",
            state: "DONE",
            members: List:Member [Member {name: "Axel", role: "Developer"}],
            tag: "Backend",
            subTasks: List:Task []
          }
        ]
      },
      Task {
        title: "Interaction mechanics",
        description: "As a developer, I want the interaction mechanics between Pac-Man and the ghosts, including losing lives when Pac-Man is touched and the ability to eat ghosts when consuming power pellets.",
        state: "DONE",
        members: List:Member [Member {name: "Sebas", role: "Developer"}],
        tag: "Backend",
        subTasks: List:Task []
      }
    ]
  }
}

// Create task for QA testing with pattern matching
func createTaskForTesting -> Task {
    params {
        titleTask: StringParagraph,
        descriptionTask: StringParagraph,
        stateTask: State,
        tag: Tag
    }
    pattern {
      case (_, _, _, "QA") {
        Task {
          title: titleTask,
          description: descriptionTask,
          state: stateTask,
          members: List:Member [Member {name: "Luiggy", role: "Developer"}],
          tag: tag,
```

```
          subTasks: List:Task []
        }
      default {
        Task {
          title: "Task for testing",
          description: "Task for testing",
          state: "NoStatus",
          members: List:Member [],
          tag: "QA",
          subTasks: List:Task []
          }
        }
      }
    }
}

// Function execution
do {
  createTaskForTesting(
    "Initial testing of Pac-Man mechanics",
    "As a developer, I want to conduct initial tests to ensure the implemented mechanics work
↪  correctly in various scenarios and there are no logical errors regarding the implemented AI
↪  logic.",
    "ToDo",
    "QA"
    )
}
```

3 SPRINT 3

```
// SPRINT 3
func sprint3Tasks -> List:Task {
  params {}
  return {
    List:Task [
      Task {
        title: "Complete level design",
        description: "As a user, I want to finalize the design and implementation of all game
↪  levels.",
        state: "DONE",
        members: List:Member [Member {name: "Sebas", role: "Designer"}],
        tag: "Frontend",
        subTasks: List:Task []
      },
```

```
      Task {
        title: "Basic UI implementation",
        description: "As a developer, I want to have the user interface developed and completed,
↪  including start screens, settings menus, and high score screens.",
        state: "DONE",
        members: List:Member [Member {name: "Luiggy", role: "Developer"}],
        tag: "UI",
        subTasks: List:Task []
      }
    ]
  }
}

// Create task for QA testing
func createTaskForTesting -> Task {
    params {
        titleTask: StringParagraph,
        descriptionTask: StringParagraph,
        stateTask: State,
        tag: Tag
    }
    return {
        Task {
          title: titleTask,
                description: descriptionTask,
                state: stateTask,
                members: List:Member [Member {name: "Luiggy", role: "Developer"}],
                tag: tag,
                subTasks: List:Task []
        }
    }
}

// Verify if the task is for QA
func verifiedIfTaskIsForQA -> Bool {
  params { myTask: Task }
  return {
    if (myTask.tag == "QA") then True
    else False
  }
}

// Function execution
do {
  verifiedIfTaskIsForQA(createTaskForTesting(
    "UI and Level Testing",
    "As a developer, I want thorough tests to ensure levels are well designed and the UI works as
↪  expected on different platforms and resolutions.",
```

```
    "ToDo",
    "QA"
    ))
}
```

4 SPRINT 4

```
// SPRINT 4
func sprint4Tasks -> List:Task {
  params {}
  return {
    List:Task [
      Task {
        title: "Sound effects",
        description: "As a developer, I want to add sound effects for Pac-Man's actions and
↪  interactions with ghosts and special points.",
        state: "DONE",
        members: List:Member [Member {name: "Santiago", role: "Developer"}],
        tag: "Backend",
        subTasks: List:Task [
          Task {
            title: "Sound effects creation",
            description: "As a developer, I want to design and develop specific sound effects for
↪  each game action.",
            state: "DONE",
            members: List:Member [Member {name: "Axel", role: "Developer"}],
            tag: "Backend",
            subTasks: List:Task []
          }
        ]
      },
      Task {
        title: "Sound integration testing",
        description: "As a developer, I want to ensure sound effects and background music are
↪  correctly integrated with the game.",
        state: "InProgress",
        members: List:Member [Member {name: "Diego", role: "QA Tester"}],
        tag: "QA",
        subTasks: List:Task []
      }
    ]
  }
}

```

```
// Create task for background music
func createTaskForBackgroundMusic -> Task {
    params {
        titleTask: StringParagraph,
        descriptionTask: StringParagraph,
        stateTask: State,
        tag: Tag
    }
    return {
        Task {
          title: titleTask,
                description: descriptionTask,
                state: stateTask,
                members: List:Member [Member {name: "Sebas", role: "Developer"}],
                tag: tag,
                subTasks: List:Task []
        }
    }
}

// Verify if the task is in progress
func verifiedIfTaskIsInProgress -> Bool {
  params { myTask: Task }
  return {
    if (myTask.tag == "InProgress") then True
    else False
  }
}

// Function execution
do {
  verifiedIfTaskIsInProgress(createTaskForBackgroundMusic(
    "Background music implementation",
    "As a developer, I want to compose and adapt background music that enhances the gaming experience
↪  without being intrusive.",
    "InProgress",
    "Backend"
    ))
}
```