

Sample language - 4 Sprints Pacman

Hufflepuff

June 17, 2024

1 SPRINT 1

```
1 // SPRINT 1
2 func sprint1Tasks -> List:Task {
3     params {}
4     return {
5         List:Task [
6             Task {
7                 title: "Development environment setup",
8                 description: "As a developer, I want to have all necessary tools like compilers, development
9 ↪ environments, and version control systems installed.",
10                state: "DONE",
11                members: List:Member [Member {name: "Axel", role: "Developer"}],
12                tag: "Backend",
13                subTasks: List:Task []
14            },
15            Task {
16                title: "Initial maze design",
17                description: "As a user, I want to create a basic maze design using graphic design tools.",
18                state: "DONE",
19                members: List:Member [Member {name: "Diego", role: "Designer"}],
20                tag: "Frontend",
21                subTasks: List:Task []
22            }
23        ]
24    }
25
26 // Create task for Pacman movement
27 func taskMovementPacman -> Task {
28     params {
29         titleTask: StringParagraph
30     }
31     return {
32         Task {
```

```

33         title: titleTask,
34         description: "As a developer, I want to implement the basic movement of Pacman in the
↪ maze.",
35         state: "DONE",
36         members: List:Member [Member {name: "Luiggy", role: "Developer"}],
37         tag: "Backend",
38         subTasks: List:Task []
39     }
40 }
41 }
42
43 // Print task titles
44 func printTasksTitle -> String {
45     params { task: Task }
46     return {
47         task.title
48     }
49 }
50
51 // Check task status
52 func taskIsDone -> Bool {
53     params { myTask: Task }
54     return {
55         if (myTask.state == "DONE") then True
56         else False
57     }
58 }
59
60 // Function execution
61 do {
62     taskIsDone(taskMovementPacman("Basic Pacman movement"))
63     // Since do can only handle one function at a time, another example of execution is:
64     map(sprint1Tasks, printTasksTitle)
65 }

```

2 SPRINT 2

```

1 // SPRINT 2
2 func sprint2Tasks -> List:Task {
3     params {}
4     return {
5         List:Task [
6             Task {
7                 title: "Spike: Research ghost movement patterns",

```

```

8         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.",
9         state: "DONE",
10        members: List:Member [Member {name: "Santiago", role: "Developer"}],
11        tag: "Spike",
12        subTasks: List:Task [
13            Task {
14                title: "Experimenting with AI algorithms",
15                description: "As a user, I want to implement AI algorithm prototypes to simulate
↪ autonomous and adaptive ghost movements.",
16                state: "DONE",
17                members: List:Member [Member {name: "Axel", role: "Developer"}],
18                tag: "Backend",
19                subTasks: List:Task []
20            }
21        ]
22    },
23    Task {
24        title: "Interaction mechanics",
25        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.",
26        state: "DONE",
27        members: List:Member [Member {name: "Sebas", role: "Developer"}],
28        tag: "Backend",
29        subTasks: List:Task []
30    }
31 ]
32 }
33 }
34
35 // Create task for QA testing with pattern matching
36 func createTaskForTesting -> Task {
37     params {
38         titleTask: StringParagraph,
39         descriptionTask: StringParagraph,
40         stateTask: State,
41         tag: Tag
42     }
43     pattern {
44         case (_, _, _, "QA") {
45             Task {
46                 title: titleTask,
47                 description: descriptionTask,
48                 state: stateTask,
49                 members: List:Member [Member {name: "Luiggy", role: "Developer"}],
50                 tag: tag,

```

```

51     subTasks: List:Task []
52   }
53   default {
54     Task {
55       title: "Task for testing",
56       description: "Task for testing",
57       state: "NoStatus",
58       members: List:Member [],
59       tag: "QA",
60       subTasks: List:Task []
61     }
62   }
63 }
64 }
65 }
66
67 // Function execution
68 do {
69   createTaskForTesting(
70     "Initial testing of Pac-Man mechanics",
71     "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.",
72     "ToDo",
73     "QA"
74   )
75 }

```

3 SPRINT 3

```

1 // SPRINT 3
2 func sprint3Tasks -> List:Task {
3   params {}
4   return {
5     List:Task [
6       Task {
7         title: "Complete level design",
8         description: "As a user, I want to finalize the design and implementation of all game
↪ levels.",
9         state: "DONE",
10        members: List:Member [Member {name: "Sebas", role: "Designer"}],
11        tag: "Frontend",
12        subTasks: List:Task []
13      },

```

```

14     Task {
15         title: "Basic UI implementation",
16         description: "As a developer, I want to have the user interface developed and completed,
↪ including start screens, settings menus, and high score screens.",
17         state: "DONE",
18         members: List:Member [Member {name: "Luigy", role: "Developer"}],
19         tag: "UI",
20         subTasks: List:Task []
21     }
22 ]
23 }
24 }
25
26 // Create task for QA testing
27 func createTaskForTesting -> Task {
28     params {
29         titleTask: StringParagraph,
30         descriptionTask: StringParagraph,
31         stateTask: State,
32         tag: Tag
33     }
34     return {
35         Task {
36             title: titleTask,
37             description: descriptionTask,
38             state: stateTask,
39             members: List:Member [Member {name: "Luigy", role: "Developer"}],
40             tag: tag,
41             subTasks: List:Task []
42         }
43     }
44 }
45
46 // Verify if the task is for QA
47 func verifiedIfTaskIsForQA -> Bool {
48     params { myTask: Task }
49     return {
50         if (myTask.tag == "QA") then True
51         else False
52     }
53 }
54
55 // Function execution
56 do {
57     verifiedIfTaskIsForQA(createTaskForTesting(
58         "UI and Level Testing",
59         "As a developer, I want thorough tests to ensure levels are well designed and the UI works as
↪ expected on different platforms and resolutions.",

```

```

60     "ToDo",
61     "QA"
62 ))
63 }

```

4 SPRINT 4

```

1 // SPRINT 4
2 func sprint4Tasks -> List:Task {
3     params {}
4     return {
5         List:Task [
6             Task {
7                 title: "Sound effects",
8                 description: "As a developer, I want to add sound effects for Pac-Man's actions and
↳ interactions with ghosts and special points.",
9                 state: "DONE",
10                members: List:Member [Member {name: "Santiago", role: "Developer"}]],
11                tag: "Backend",
12                subTasks: List:Task [
13                    Task {
14                        title: "Sound effects creation",
15                        description: "As a developer, I want to design and develop specific sound effects for
↳ each game action.",
16                        state: "DONE",
17                        members: List:Member [Member {name: "Axel", role: "Developer"}]],
18                        tag: "Backend",
19                        subTasks: List:Task []
20                    }
21                ]
22            },
23            Task {
24                title: "Sound integration testing",
25                description: "As a developer, I want to ensure sound effects and background music are
↳ correctly integrated with the game.",
26                state: "InProgress",
27                members: List:Member [Member {name: "Diego", role: "QA Tester"}]],
28                tag: "QA",
29                subTasks: List:Task []
30            }
31        ]
32    }
33 }
34

```

```

35 // Create task for background music
36 func createTaskForBackgroundMusic -> Task {
37     params {
38         titleTask: StringParagraph,
39         descriptionTask: StringParagraph,
40         stateTask: State,
41         tag: Tag
42     }
43     return {
44         Task {
45             title: titleTask,
46             description: descriptionTask,
47             state: stateTask,
48             members: List:Member [Member {name: "Sebas", role: "Developer"}],
49             tag: tag,
50             subTasks: List:Task []
51         }
52     }
53 }
54
55 // Verify if the task is in progress
56 func verifiedIfTaskIsInProgress -> Bool {
57     params { myTask: Task }
58     return {
59         if (myTask.tag == "InProgress") then True
60         else False
61     }
62 }
63
64 // Function execution
65 do {
66     verifiedIfTaskIsInProgress(createTaskForBackgroundMusic(
67         "Background music implementation",
68         "As a developer, I want to compose and adapt background music that enhances the gaming experience
↳ without being intrusive.",
69         "InProgress",
70         "Backend"
71     ))
72 }

```