**Report on** Youth Hockey Cup (A Non-Profit Project) **Team Management Program using Python.**

**Author – Ajay D Nair**

**Completion\_date = 2024-10-30**

**Table of Contents**

* Introduction
* Description of the Solution
* Team Class
* User interface Class
* User guide
* Conclusion

**Introduction**

The Youth Hockey Cup Team Management Program is a software application designed to manage team registrations and administration for a youth hockey cup. The purpose of this program is to facilitate the easy addition, management, and record-keeping of participating teams. As this cup is being organized for charitable purposes, the fee for each team is flexible, allowing them to contribute as much as they wish. All proceeds from the registration fees will be donated to support the education of homeless children. This program provides a user-friendly interface for organizers to create, update, list, and delete teams, along with an option to generate reports on the teams’ payment status.

**Description of the Solution:**

The solution is organized into two primary classes: **Team** and **User\_interface**, each serving specific functions in the program.

**Team Class**

The **Team** class in class\_team.py is responsible for encapsulating the details and operations related to each hockey team.

* **Attributes**:
  + **name:** The name of the team.
  + **type:** The type of team ("boy" or "girl").
  + **fee:** The fee paid by the team(Whatever the amount they wish to pay as donation).
  + **fee\_paid:** A boolean indicating if the fee has been paid.
  + **cancel\_date:** Optional, indicating if and when the team was canceled.
  + **date:** This is automatically created whenever a team is added and is immutable
  + **id:** It is the unique id which is generated whenever a team is added. Here it starts from 1
* **Methods**:
  + **Constructor**: Initializes a new team with attributes, including auto-generating a unique id and setting the creation date.
  + **Setters and Getters**: Accessor and mutator methods allow controlled access to each attribute.
  + **String Representation (\_\_str\_\_)**: Provides a readable output summarizing the team details, including payment and cancellation information.
* **Responsibilities**:
  + Creating a team object with essential details.
  + Managing team attributes, including setting fees and canceling teams if needed.
  + Providing easy access to team details through the str representation.

The **Team** class operates independently of other classes and serves as a data model for each team entity in the program.

**User\_interface Class**

The **User\_interface** class, defined in team\_database.py, manages interactions with the user and provides functions to manage the list of teams.

* **Attributes**:
  + **self.\_\_teams:** A private attribute list that stores all team objects created during the session.
* **Methods**:
  + **create()**: Adds a new team object to the list.
  + **delete()**: Removes a team based on a unique team ID.
  + **read()**: Returns details of a team using its ID.
  + **list\_teams()**: Lists all teams of a specific type ("boy" or "girl").
  + **list\_all()**: Lists all registered teams.
  + **total()**: Calculates and displays the total number of teams and the percentage of those who have paid their fees.
  + **update()**: Allows for updating team attributes, such as name, type, fee status, and cancellation date.
  + **download\_txt\_file**: Allows for downloading and storing information on teams in a text file for use later.
  + **restore\_data\_txt**: Allows for restoring data to the program from a text file which had been previously stored.
* **Responsibilities**:
  + Coordinating the actions performed on Team objects.
  + Maintaining a dynamic list of teams that can be updated, deleted, or listed.
  + Calculating aggregate statistics, such as the number and payment status of teams.
  + Downloading, storing data in a text file and then restoring data. Text file being a permanent data storage space.
* **Collaborations**:
  + The **User\_interface** class interacts extensively with the **Team** class, using instances of Team to perform its functions, from creation and deletion to updating and listing.

**User Guide**

The program can be run by executing the team\_database.py file. Follow these steps:

* Run the Program: Execute team\_database.py in a Python environment.
* Menu Navigation:
  + Upon launch, the program displays a menu with options to add, view, update, or delete teams.
  + Use numeric input to select a menu option.
* Adding a New Team:
  + Select option 1.
  + Enter details such as team name, type= should be “boy” or “girl” , fee, and payment status.
* Viewing a Team:
  + Use option 2 to view details by entering a team ID.
* Listing Teams by Type:
  + Select option 3 and specify "boy" or "girl" to filter teams by type.
* Updating Team Information:
  + Choose option 5 and enter the team ID. Follow prompts to select and modify attributes such as type, name, fee paid and adding cancellation date.
* Deleting a Team:
  + Option 6 allows deletion by entering a team ID.
* Viewing Payment Statistics:
  + Option 7 displays the number of registered teams and the percentage that have paid fees.
* Downloading data into text file:
  + Option 8 allows us to download the teams data into a text file and store there.
* Restoring data from text file:
  + Option 9 allows us to restore data from the previously saved text file. Allowing us to work on that. The data restored will not have the original id’s and date as those attributes are immutable.
* Exit:
  + Option 10 allows us to exit from the program

**Conclusion**

This program provides a streamlined, object-oriented solution for managing teams in the Youth Hockey Cup. Through a simple interface and class-based structure, it enables organizers to efficiently oversee team registrations and finances, while keeping their focus on the primary goal: supporting the education of homeless children through the cup proceeds.