Republic of the Philippines
SURIGAO DEL NORTE STATE UNIVERSITY
Narciso Street, Surigao City 8400, Philippines

In Partial Fulfillment of the Requirements for the
CS 223 – Object – Oriented Programming

“Four Principles of Object – Oriented Programming”

Presented to:

Dr. Unife O. Cagas
Professor

Prepared by:

Narciso, Jamby E.
BSCS 2A2

"Genre-Specific Movie Selection and Feedback System: A Python Implementation"

Title

DESCRIPTION

This project is a genre-specific movie selection and feedback system in Python. The system offers a user a selected choice of movies, broadly classified into three genres: Comedy, Action, and Drama. Each genre is represented by a corresponding Python class that will allow the instantiation of movie objects with particular attributes, such as title, duration, director, and year of release. Users of the system interact via a command-line interface whereby they pick a movie to view and give their reactions to the viewing experience. When a movie is selected, the system plays the chosen movie and asks the user to give reactions to it. The user is expected to react with categorizations such as "funny," "sad," or "satisfied." The system responds to user feedback and advances user interaction by giving an appropriate response. This implementation aims at understanding user preferences across different movie genres while making the experience of watching movies interactive.

OBJECTIVES

The following outline of the specific aims of the research in this genre-specific movie selection and feedback system in Python:

Offer users the selection of movies in comedy, action, and drama genres. Allow interaction of users through the command line interface. Make classes for each movie genre to organize and instantiate movie objects easily. Offer users the functionality of selecting a movie, viewing its details, and giving feedback regarding the movie being viewed. Enhance the engagement of users through the appreciation and reaction to user feedback appropriately. Understand the preferences and reactions of users to different movie genres for insight into viewer behavior and preference.

IMPORTANCE OR CONTRIBUTION OF THE CODE

The genre-specific movie selection and feedback code promises immense contributions:

It provides curated movie choices and interactive feedback mechanisms to improve the general view of a movie. It offers efficient handling and easy scalability of movie objects by being genre-specific through the classes. It contributes to the understanding of viewer tastes and behavior through the collection of user reactions. It provides real-world applications of object-oriented programming in Python. Through the acknowledgment of feedback, it ensures that movie-watching is dynamic and interactive.

Essentially, the code enhances movie selection processes, advances user engagement, and gives valuable insights into audience preferences.

4 PRINCIPLES OF OOP

ENCAPSULATION

```
class Movie:
    def __init__(self, title, genre, duration, director=None,
release_year=None):
        self.title = title
        self.genre = genre
        self.duration = duration
        self.director = director
        self.release_year = release_year

    def play(self):
        print(f"Playing {self.genre} movie: {self.title} ({self.duration}
mins)")
        if self.director:
            print(f"Directed by: {self.director}")
        if self.release_year:
            print(f"Released: {self.release_year}")

    def details(self):
        print(f"{self.genre}: {self.title} - {self.duration} mins")
        if self.director:
            print(f"Directed by: {self.director}")
        if self.release_year:
            print(f"Released: {self.release_year}")
```

These attributes—title, genre, duration, director, and release_year—are encapsulated within the class. They are accessed and modified using the methods, such as __init__(), play(), and details().

INHERITANCE

```
class Comedy(Movie):
    def __init__(self, title, duration, director=None, release_year=None):
        super().__init__(title, "Comedy", duration, director, release_year)

    def laugh_track(self):
        print("THANK YOU FOR YOUR FEEDBACK")

class Action(Movie):
    def __init__(self, title, duration, director=None, release_year=None):
        super().__init__(title, "Action", duration, director, release_year)
```

```
    def display_explosions(self):
        print("THANK YOU FOR YOUR FEEDBACK")

class Drama(Movie):

   def __init__(self, title, duration, director=None, release_year=None):
        super().__init__(title, "Drama", duration, director, release_year)

    def display_tears(self):
        print("THANK YOU FOR YOUR FEEDBACK")
```

Each subclass (Comedy, Action, Drama) is derived from the Movie class. They inherit the attributes from the Movie class: title, genre, duration, director, and release_year. They also inherit the methods: play() and details().

POLYMORPHISM

```
def play_movie(movie):
    movie.play()
    reaction = input("What did you think of the movie? (Choose: 'funny',
'sad', 'satisfied'): ").lower()
    if reaction == 'funny':
        print("THANK YOU FOR YOUR FEEDBACK")
    elif reaction == 'sad':
        print("THANK YOU FOR YOUR FEEDBACK")
    elif reaction == 'satisfied':
        print("THANK YOU FOR YOUR FEEDBACK")
    else:
        print("Invalid choice.")
```

The function play_movie() takes a movie object as its parameter. The function calls play() on the movie object, which is a method defined in the Movie class. Although the play_movie() function is defined to take a Movie object, it could actually process an instance of Comedy, Action, or Drama subclasses. Objects from subclasses inherit from the Movie class, so they can be treated as if they were objects of class Movie. Thus, polymorphism is achieved, using the same function to play and interface with the various types of movies.

ABSTRACTION

```
def choose_movie():
    print("Choose a movie to watch:")
    print("1. Comedy - Home Alone")
    print("2. Action - Damsel")
    print("3. Drama - A Silent Voice")
    print("0. Exit")
    choice = input("Enter your choice (1, 2, 3, or 0 to exit): ")
    return choice

def play_movie(movie):
    movie.play()
```

```
    reaction = input("What did you think of the movie? (Choose: 'funny',
'sad', 'satisfied'): ").lower()
    if reaction == 'funny':
        print("THANK YOU FOR YOUR FEEDBACK")
    elif reaction == 'sad':
        print("THANK YOU FOR YOUR FEEDBACK")
    elif reaction == 'satisfied':
        print("THANK YOU FOR YOUR FEEDBACK")
    else:
        print("Invalid choice.")
```

The function choose_movie() gives the user a list of movie options and prompts the user to make a choice. Users interact with this function without needing to know how the movie options are presented or processed.

The function play_movie() interacts with the object Movie to play it and collect user feedback. Users input their feedback through this function without needing to understand the intricacies of movie playback or feedback processing.

Hardware & Software Used

Hardware

- Laptop

Software

- Visual Studio Code
- Online GDB

OUTPUT:

```
PS C:\Users\Toshiba\Documents\vs> & C:/Users/Toshiba/anaconda3/python.exe c:/Users/Toshiba/Documents/vs/oop.py
Choose a movie to watch:
1. Comedy - Home Alone
2. Action - Damsel
3. Drama - A Silent Voice
0. Exit
Enter your choice (1, 2, 3, or 0 to exit): 1
Playing Comedy movie: Home Alone (143 mins)
Directed by: Chris Columbus
Released: 1990
What did you think of the movie? (Choose: 'funny', 'sad', 'satisfied'): FUNNY
THANK YOU FOR YOUR FEEDBACK
Choose a movie to watch:
1. Comedy - Home Alone
2. Action - Damsel
3. Drama - A Silent Voice
0. Exit
Enter your choice (1, 2, 3, or 0 to exit): 0
Goodbye!
PS C:\Users\Toshiba\Documents\vs>
```

They select a movie by entering the corresponding number. The program plays the movie, showing the details of the title, duration, director, and year of release. Then it asks the user for feedback: was the movie 'funny', 'sad', or 'satisfied' with the movie. The program greets the feedback, and the program continues running until the '0' button is pressed for exit. Upon exiting, the program greets the user with "Goodbye!" to signal the end of the interaction.

CODE:

```
class Movie:
    def __init__(self, title, genre, duration, director=None,
release_year=None):
        # Initializes a Movie object with title, genre, duration, director,
and release year.
        self.title = title
        self.genre = genre
        self.duration = duration
        self.director = director
        self.release_year = release_year

    def play(self):
        # Displays a message indicating the movie is being played, including
genre, title, and duration.
        print(f"Playing {self.genre} movie: {self.title} ({self.duration}
mins)")
        # If director is provided, displays the director's name.
        if self.director:
            print(f"Directed by: {self.director}")
        # If release year is provided, displays the release year.
        if self.release_year:
            print(f"Released: {self.release_year}")

    def details(self):
        # Displays details of the movie, including genre, title, duration,
director, and release year.
        print(f"{self.genre}: {self.title} - {self.duration} mins")
        # If director is provided, displays the director's name.
        if self.director:
            print(f"Directed by: {self.director}")
        # If release year is provided, displays the release year.
        if self.release_year:
            print(f"Released: {self.release_year}")

class Comedy(Movie):
    def __init__(self, title, duration, director=None, release_year=None):
        # Initializes a Comedy object with title, duration, director, and
release year, inheriting genre from Movie.
        super().__init__(title, "Comedy", duration, director, release_year)

    def laugh_track(self):
        # Displays a message thanking for feedback specifically for comedy
movies.
        print("THANK YOU FOR YOUR FEEDBACK")

class Action(Movie):
    def __init__(self, title, duration, director=None, release_year=None):
```

```
        # Initializes an Action object with title, duration, director, and
release year, inheriting genre from Movie.
        super().__init__(title, "Action", duration, director, release_year)

    def display_explosions(self):
        # Displays a message thanking for feedback specifically for action
movies.
        print("THANK YOU FOR YOUR FEEDBACK")

class Drama(Movie):
    def __init__(self, title, duration, director=None, release_year=None):
        # Initializes a Drama object with title, duration, director, and
release year, inheriting genre from Movie.
        super().__init__(title, "Drama", duration, director, release_year)

    def display_tears(self):
        # Displays a message thanking for feedback specifically for drama
movies.
        print("THANK YOU FOR YOUR FEEDBACK")

def choose_movie():
    # Displays a menu for choosing a movie to watch.
    print("Choose a movie to watch:")
    print("1. Comedy - Home Alone")
    print("2. Action - Damsel")
    print("3. Drama - A Silent Voice")
    print("0. Exit")
    # Prompts the user to enter their choice.
    choice = input("Enter your choice (1, 2, 3, or 0 to exit): ")
    return choice

def play_movie(movie):
    # Plays the selected movie and collects user feedback.
    movie.play()
    # Prompts the user to provide feedback on the movie.
    reaction = input("What did you think of the movie? (Choose: 'funny',
'sad', 'satisfied'): ").lower()
    if reaction == 'funny' or reaction == 'sad' or reaction == 'satisfied':
        # Acknowledges the user's feedback.
        print("THANK YOU FOR YOUR FEEDBACK")
    else:
        # Displays an error message for an invalid feedback choice.
        print("Invalid choice.")

def main():
    while True:
        # Continuously prompts the user to choose a movie until they choose to
exit.
```

```
        choice = choose_movie()

        if choice == '0':
            # Exits the program if the user chooses '0'.
            print("Goodbye!")
            break

        elif choice == '1':
            # Plays the comedy movie "Home Alone" if the user chooses '1'.
            movie = Comedy("Home Alone", 143, "Chris Columbus", 1990)
            play_movie(movie)

        elif choice == '2':
            # Plays the action movie "Damsel" if the user chooses '2'.
            movie = Action("Damsel", 150, "Juan Carlos Fresnadillo", 2024)
            play_movie(movie)

        elif choice == '3':
            # Plays the drama movie "A Silent Voice" if the user chooses '3'.
            movie = Drama("A Silent Voice", 295, "Naoko Yamada", 2016)
            play_movie(movie)

        else:
            # Displays an error message for an invalid choice.
            print("Invalid choice. Please choose a valid number.")

if __name__ == "__main__":
    # Executes the main function when the script is run.
    main()
```

USER GUIDE:

Upon running the program, you will be presented with a menu to choose a movie to watch.
You can select a movie by entering the corresponding number.
To exit the program, enter '0' at any time.

```
Choose a movie to watch:
1. Comedy - Home Alone
2. Action - Damsel
3. Drama - A Silent Voice
0. Exit
Enter your choice (1, 2, 3, or 0 to exit): 1
```

Choose the movie you want to watch by entering the corresponding number.
You can select from Comedy, Action, and Drama genres.

You will see information about the movie, including its title, genre, duration, director (if available), and release year (if available).

```
Playing Comedy movie: Home Alone (143 mins)
Directed by: Chris Columbus
Released: 1990
```

You will be prompted to provide feedback.
Choose from the provided options: 'funny' (for Comedy), 'sad' (for Drama), or 'satisfied' (for Action).
Your feedback helps us improve our movie selection and enhance your viewing experience.

```
What did you think of the movie? (Choose: 'funny', 'sad', 'satisfied'): funny
THANK YOU FOR YOUR FEEDBACK
```

You can exit the program at any time by choosing '0' from the menu.
Upon exiting, you will see a "Goodbye!" message.

```
Choose a movie to watch:
1. Comedy - Home Alone
2. Action - Damsel
3. Drama - A Silent Voice
0. Exit
Enter your choice (1, 2, 3, or 0 to exit): 0
Goodbye!
```

REFERENCE

https://www.youtube.com/watch?v=ZDa-Z5JzLYM

https://www.w3schools.com/python/python_classes.asp

https://www.w3schools.com/python/python_inheritance.asp

https://www.w3schools.com/python/python_polymorphism.asp