# Terminal Application

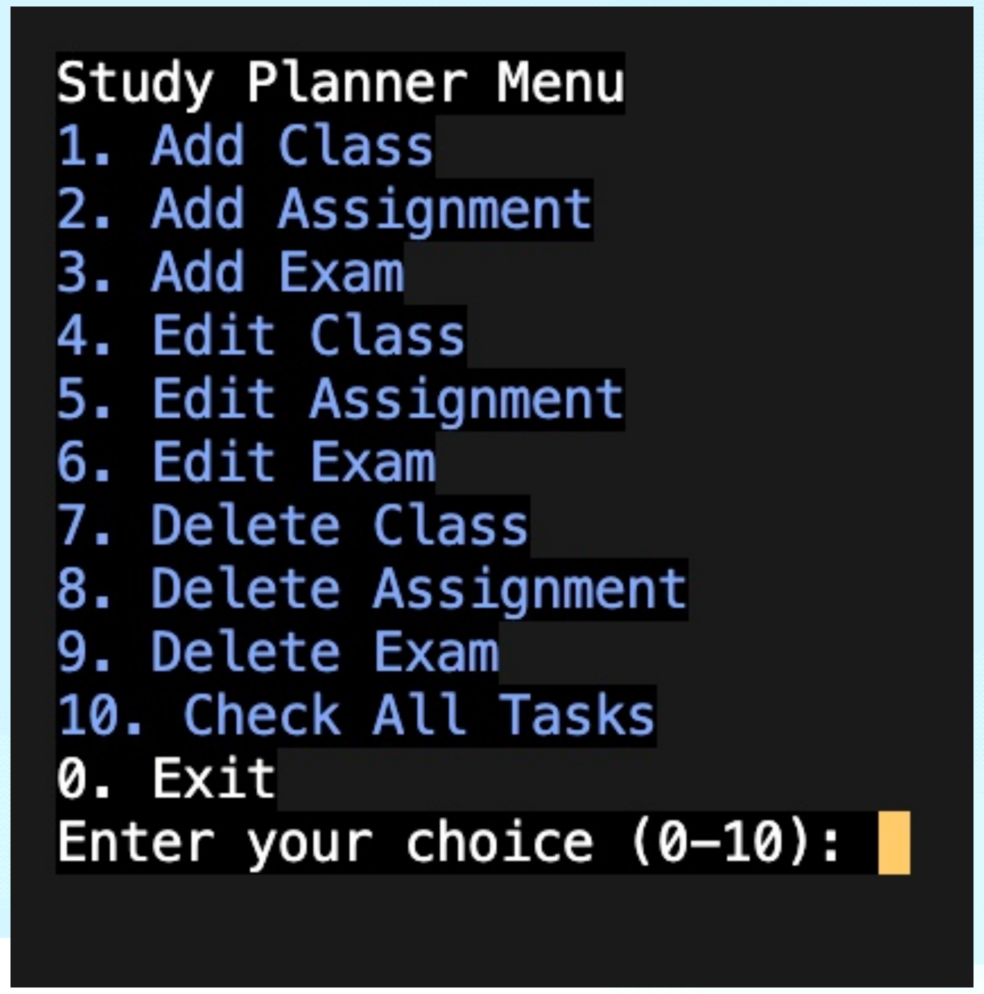
## Study Planner

Zac_Xenitopoulos_T1A3

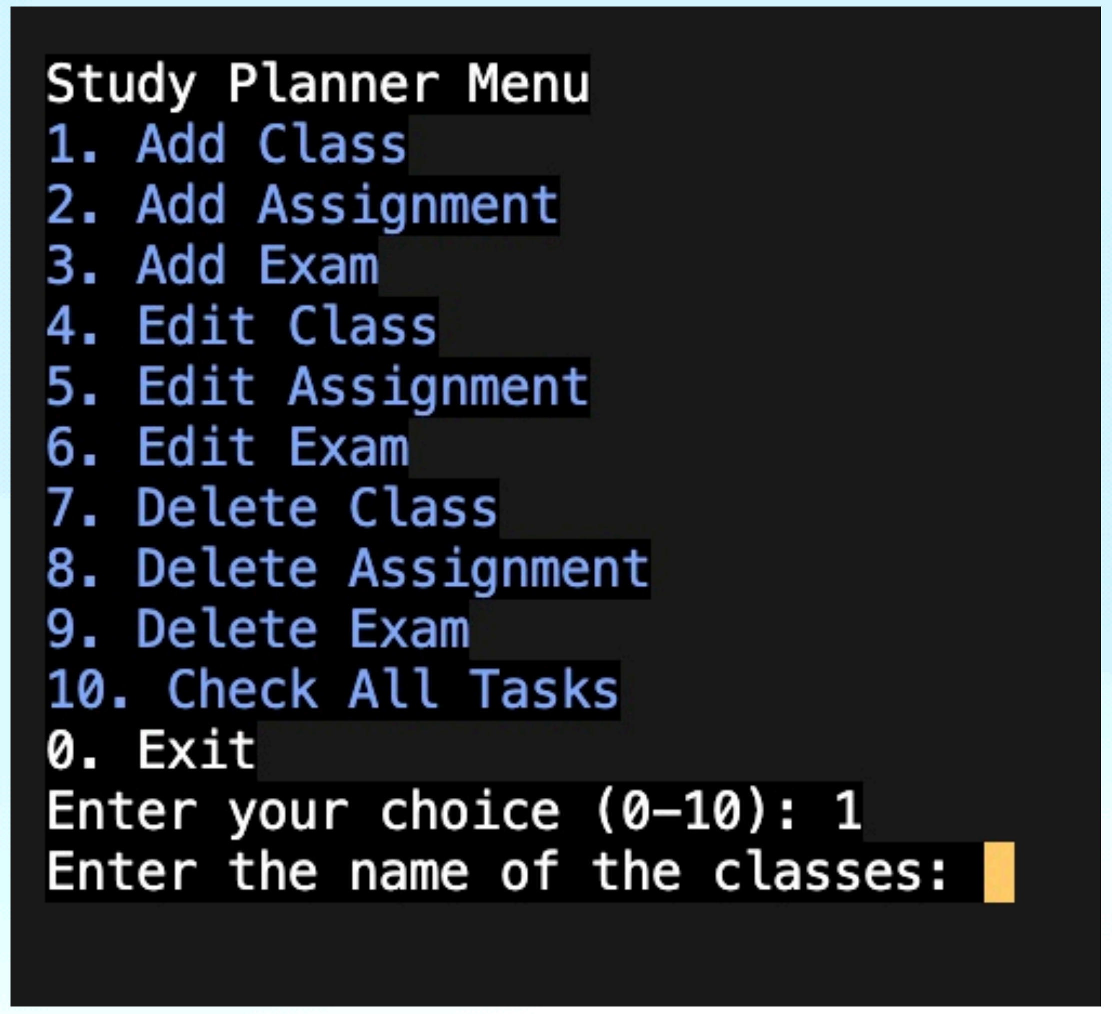
## Main Application Features

- A simple yet effective study planner application which keeps track of all upcoming classes, exams and assignments. The application will allow users to add, edit, delete and check for tasks within the application.

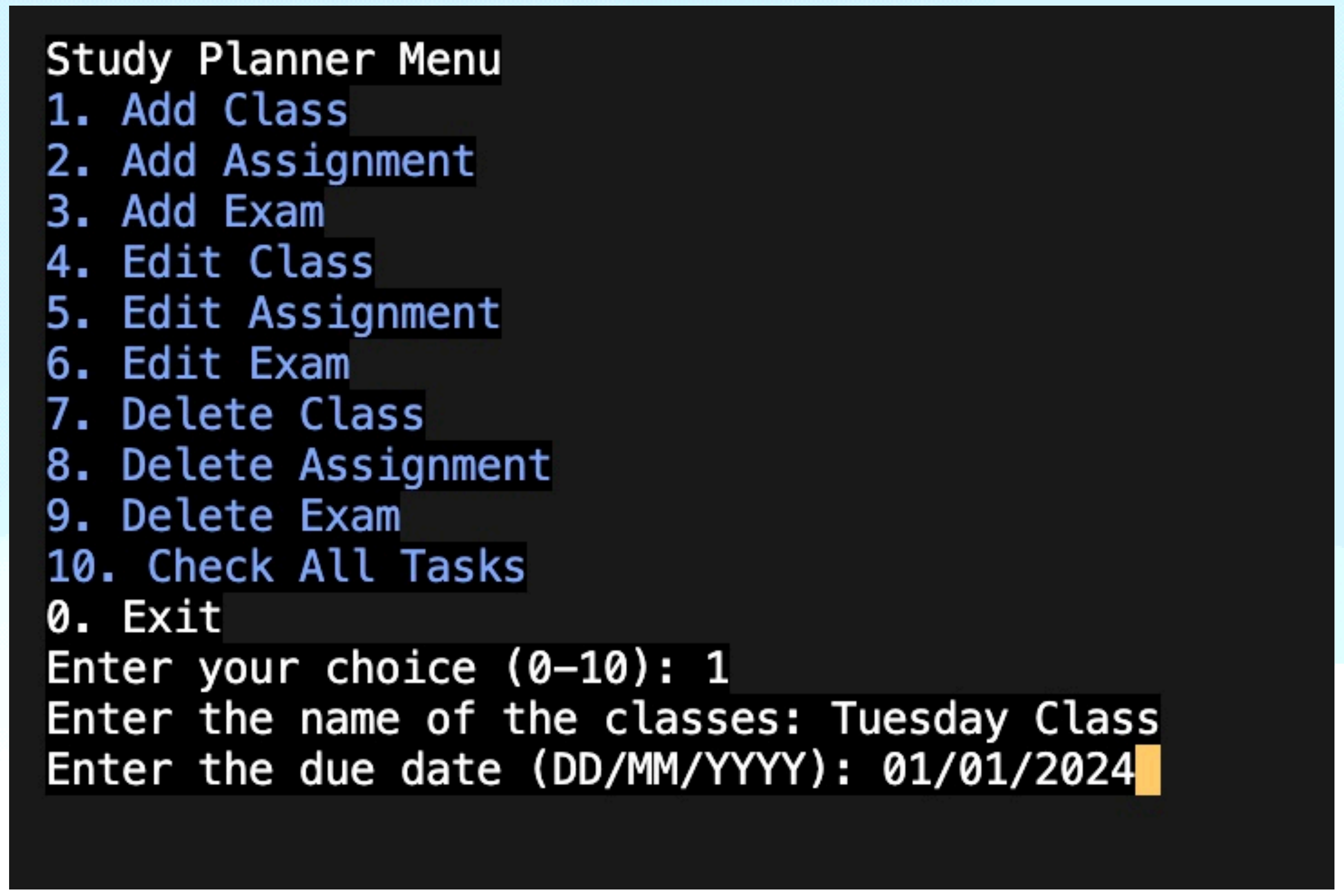
# Add task (class, assignment or exam)



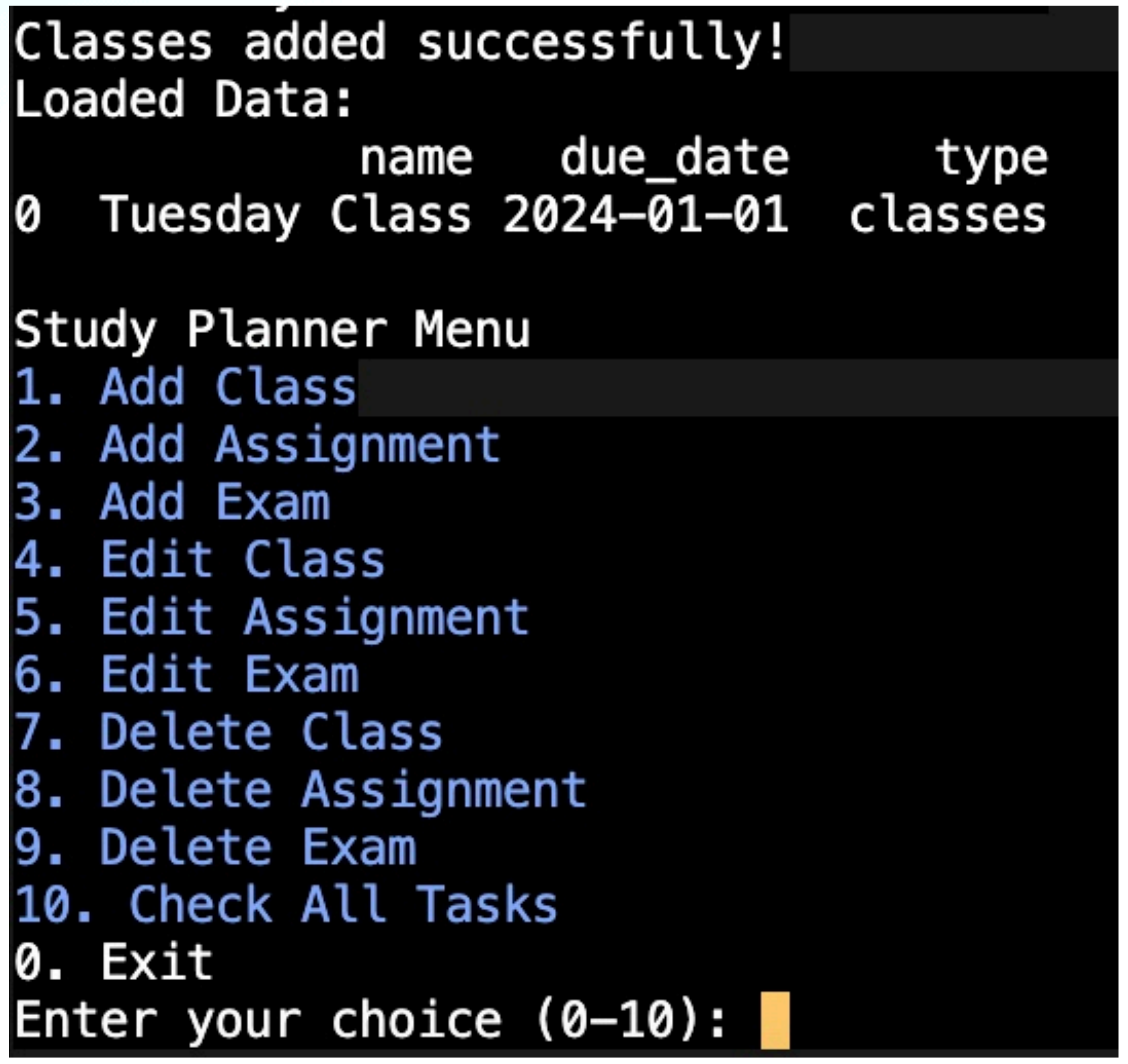
```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10):
```



```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 1
Enter the name of the classes:
```

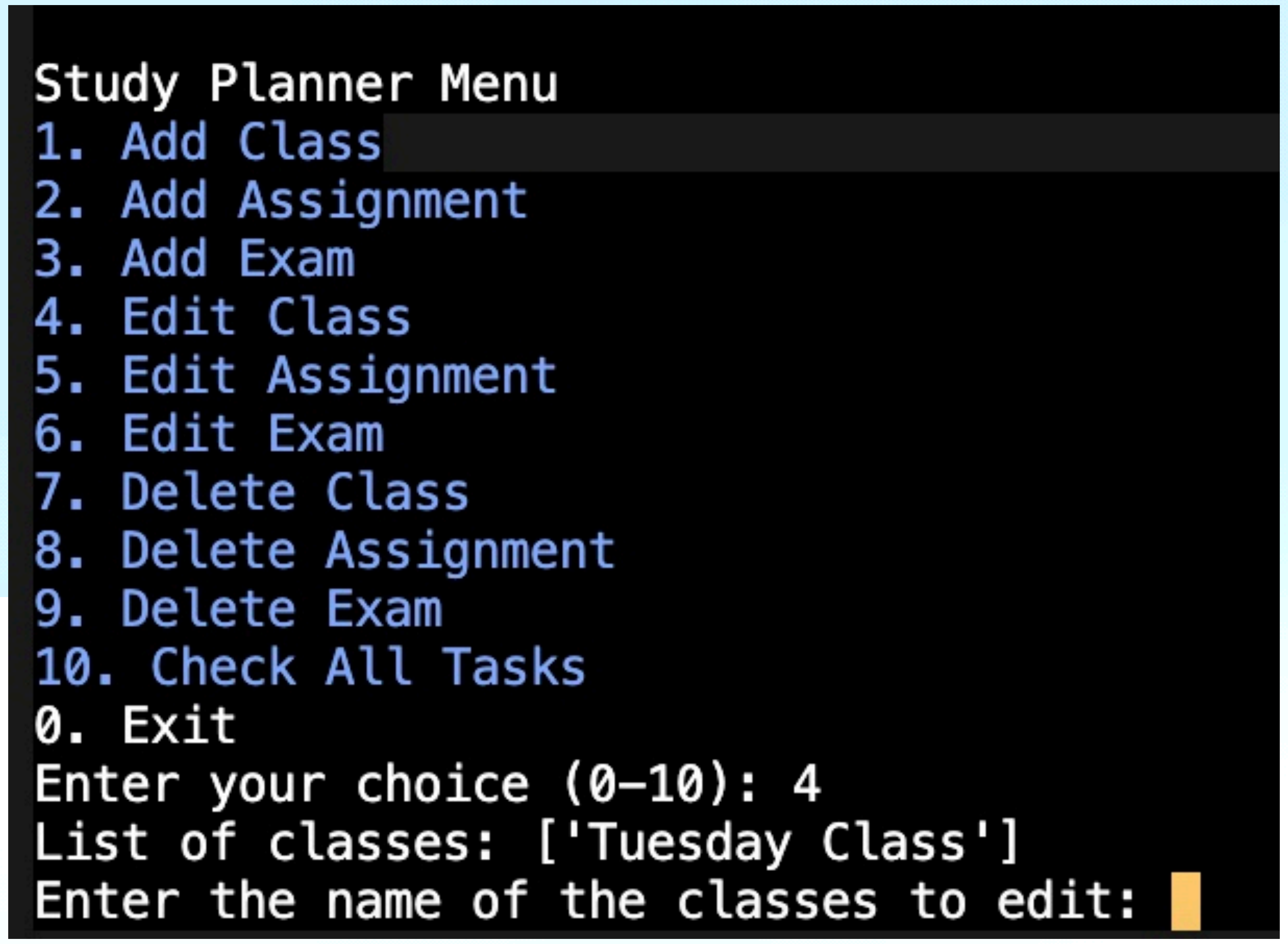


```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 1
Enter the name of the classes: Tuesday Class
Enter the due date (DD/MM/YYYY): 01/01/2024
```

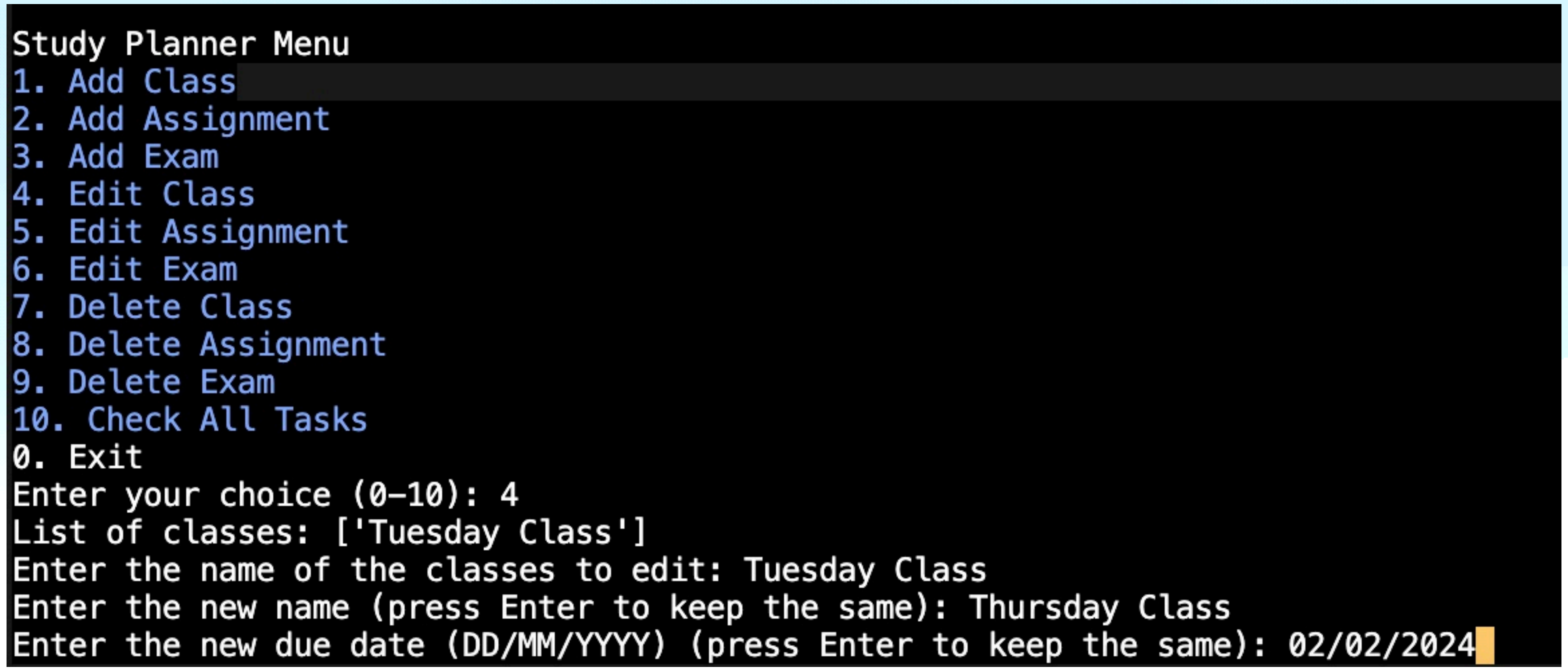


```
Classes added successfully!
Loaded Data:
            name   due_date     type
0  Tuesday Class 2024-01-01  classes

Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10):
```

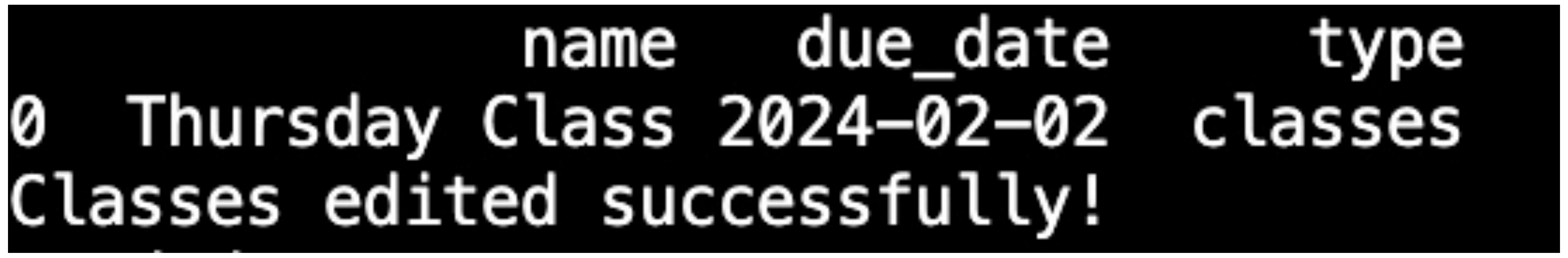
# Edit Task (classes, assignments and exams)



```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 4
List of classes: ['Tuesday Class']
Enter the name of the classes to edit: 
```

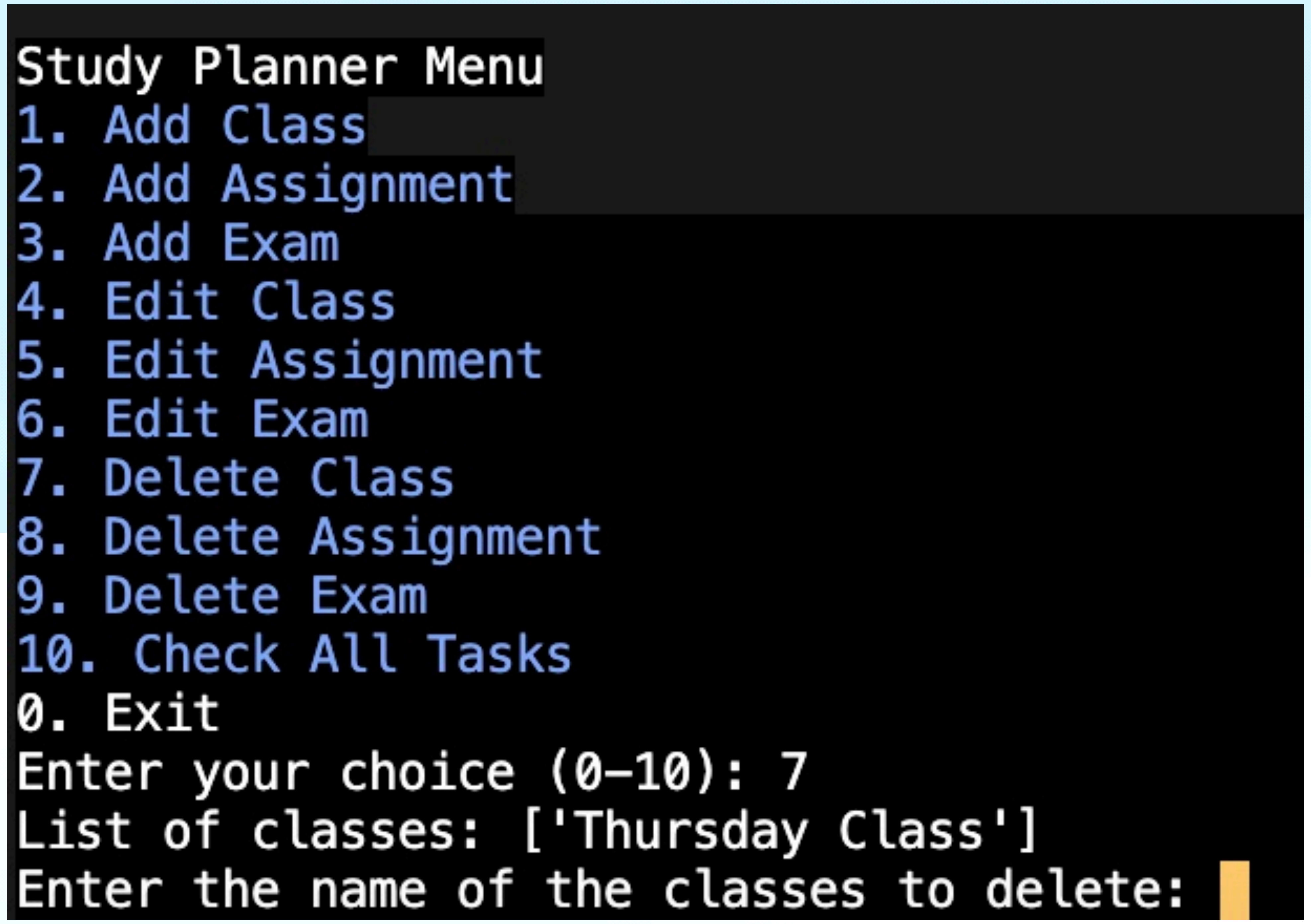


```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 4
List of classes: ['Tuesday Class']
Enter the name of the classes to edit: Tuesday Class
Enter the new name (press Enter to keep the same): Thursday Class
Enter the new due date (DD/MM/YYYY) (press Enter to keep the same): 02/02/2024
```

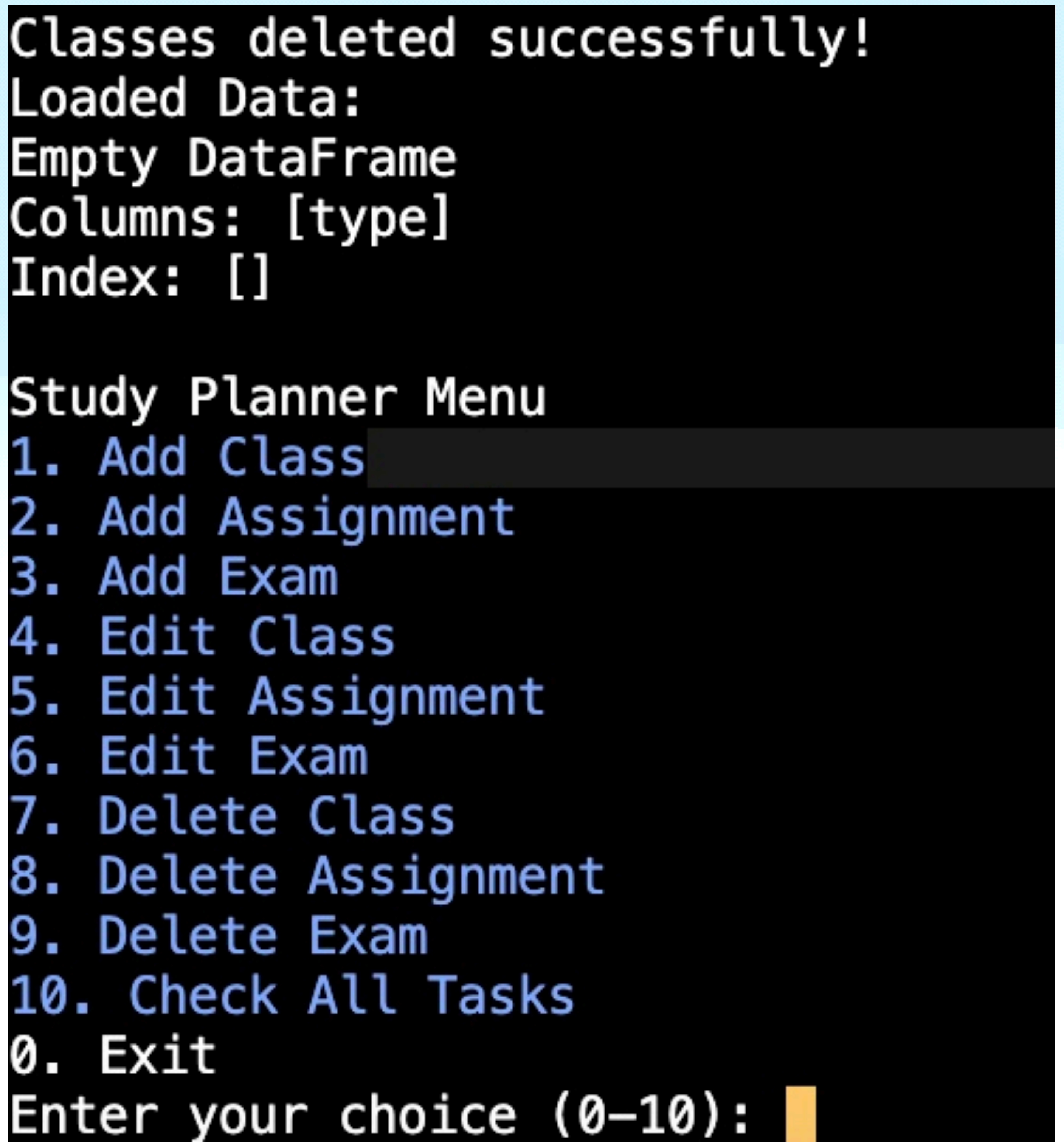


```
             name    due_date     type
0  Thursday Class 2024-02-02   classes
Classes edited successfully!
```

# Delete tasks (classes, assignments and exams)

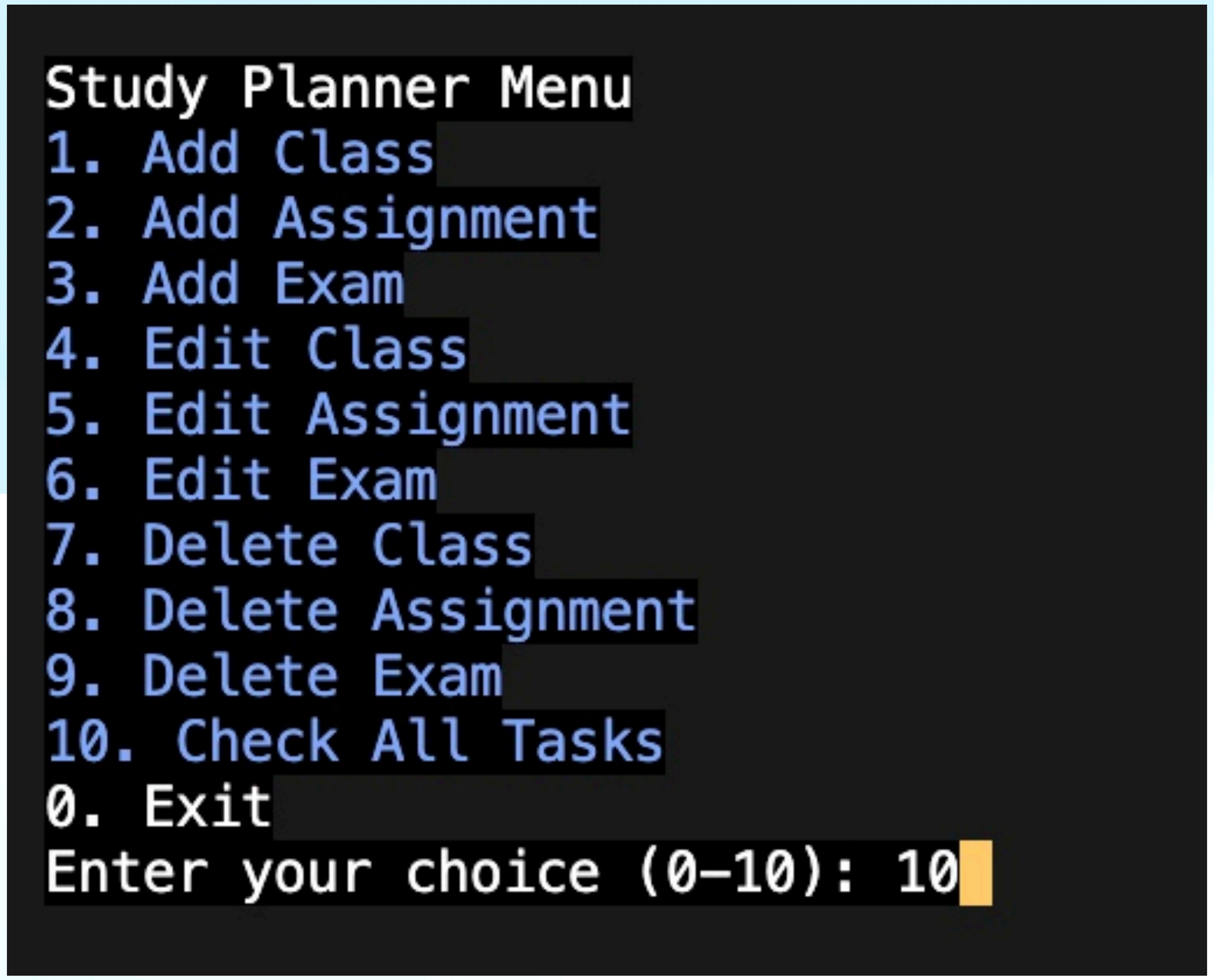


```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 7
List of classes: ['Thursday Class']
Enter the name of the classes to delete:
```

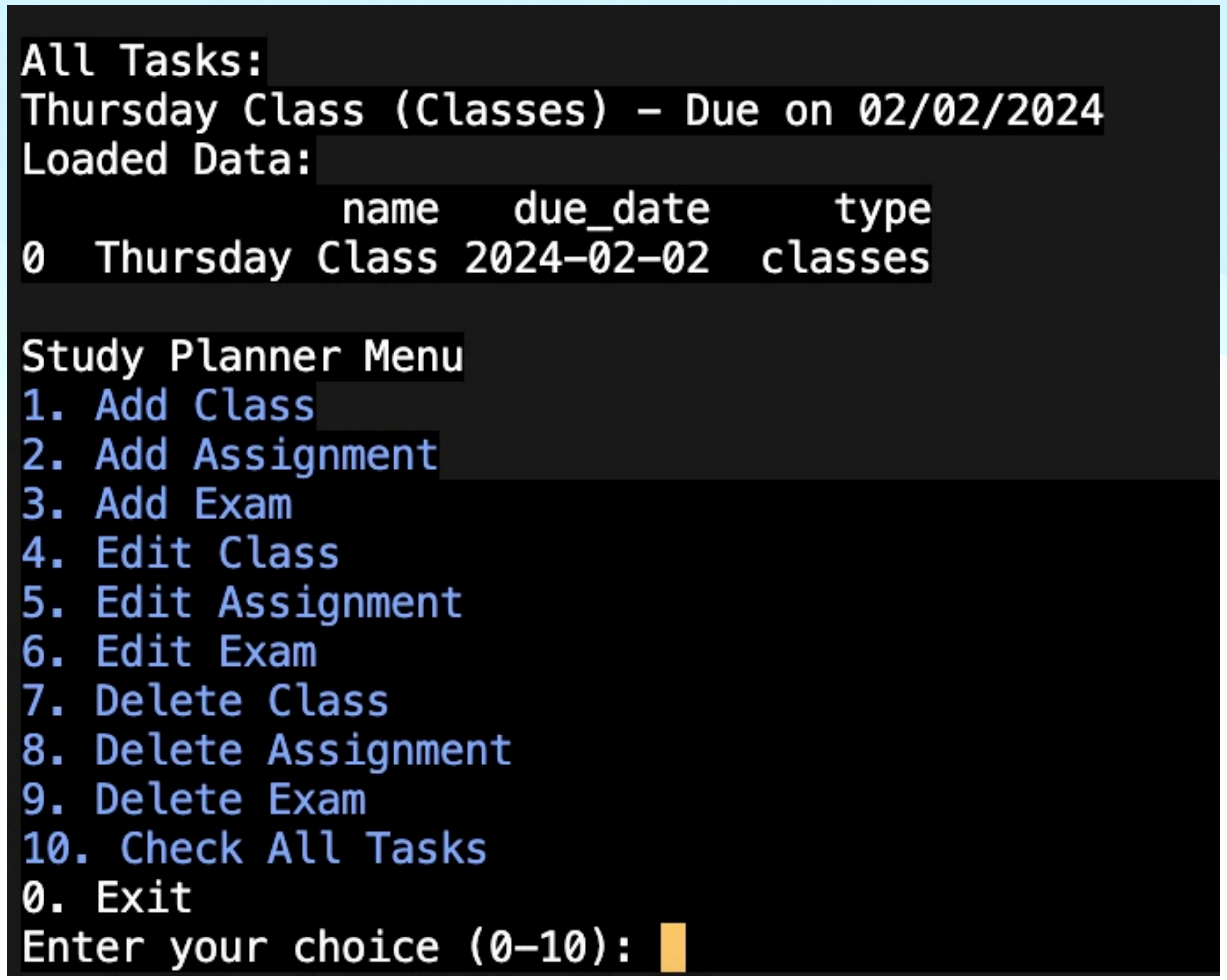


```
Classes deleted successfully!
Loaded Data:
Empty DataFrame
Columns: [type]
Index: []

Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10):
```

# Check tasks (classes, assignments and exams)



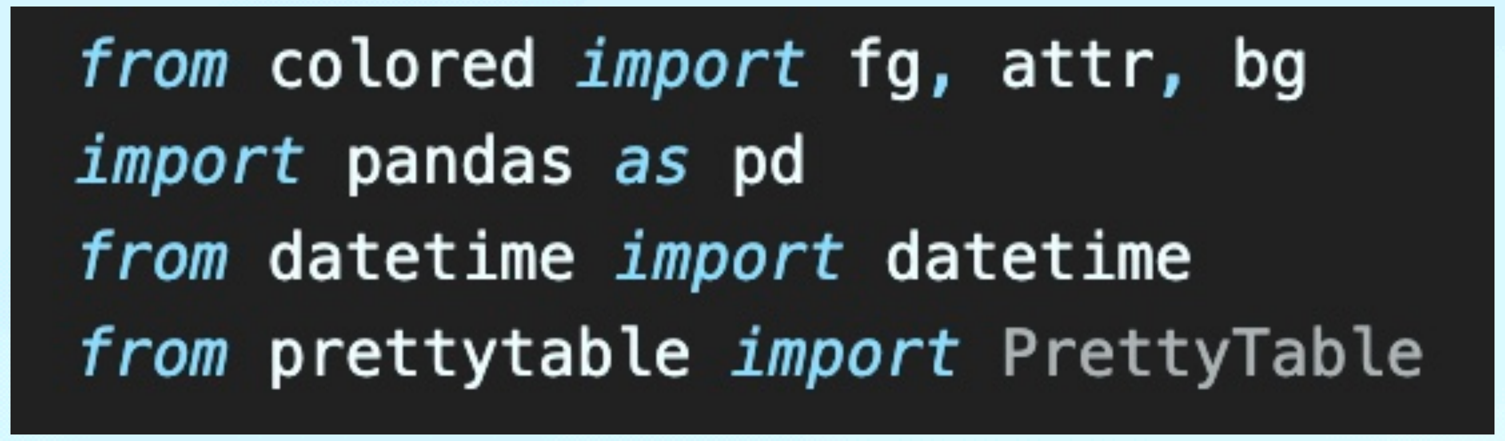
```
Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 10
```



```
All Tasks:
Thursday Class (Classes) - Due on 02/02/2024
Loaded Data:
             name    due_date     type
0  Thursday Class 2024-02-02  classes

Study Planner Menu
1. Add Class
2. Add Assignment
3. Add Exam
4. Edit Class
5. Edit Assignment
6. Edit Exam
7. Delete Class
8. Delete Assignment
9. Delete Exam
10. Check All Tasks
0. Exit
Enter your choice (0-10): 
```

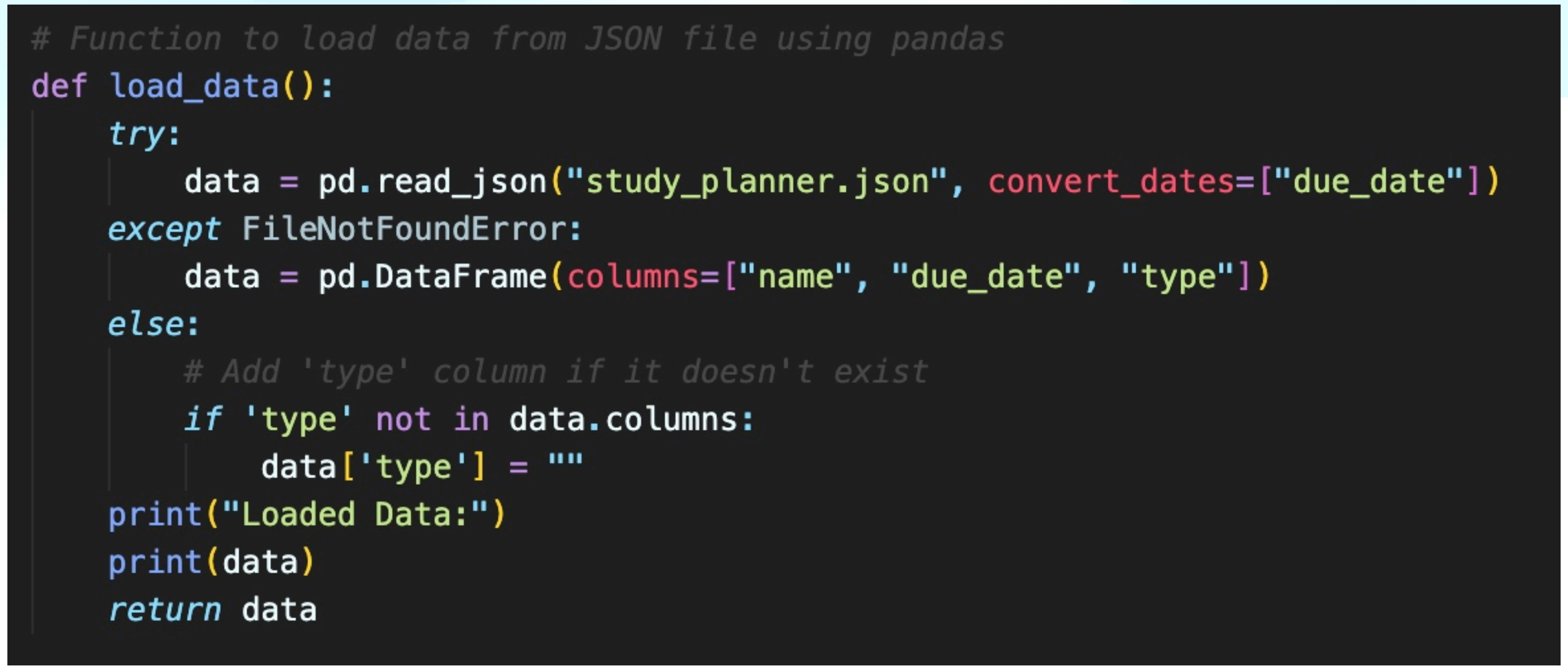
# Code Overview

## Important Statements



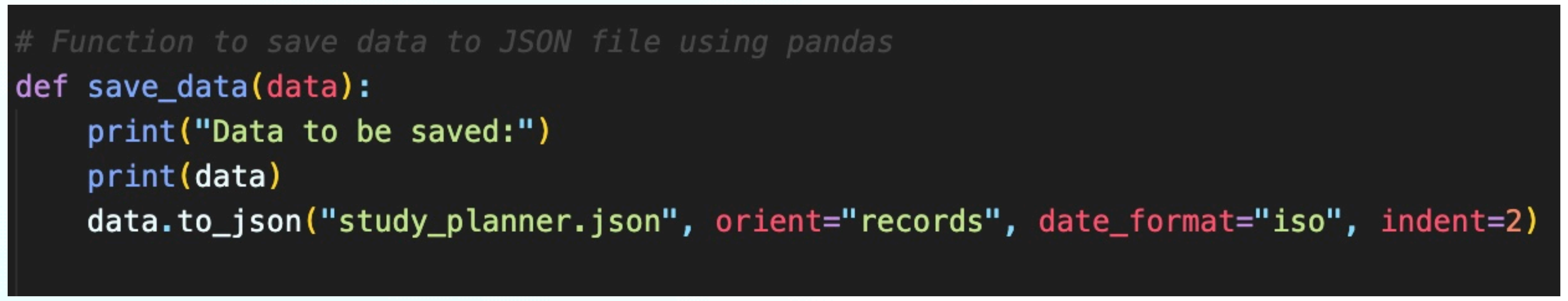
```python
from colored import fg, attr, bg
import pandas as pd
from datetime import datetime
from prettytable import PrettyTable
```

## Load Data



```python
# Function to load data from JSON file using pandas
def load_data():
    try:
        data = pd.read_json("study_planner.json", convert_dates=["due_date"])
    except FileNotFoundError:
        data = pd.DataFrame(columns=["name", "due_date", "type"])
    else:
        # Add 'type' column if it doesn't exist
        if 'type' not in data.columns:
            data['type'] = ""
    print("Loaded Data:")
    print(data)
    return data
```

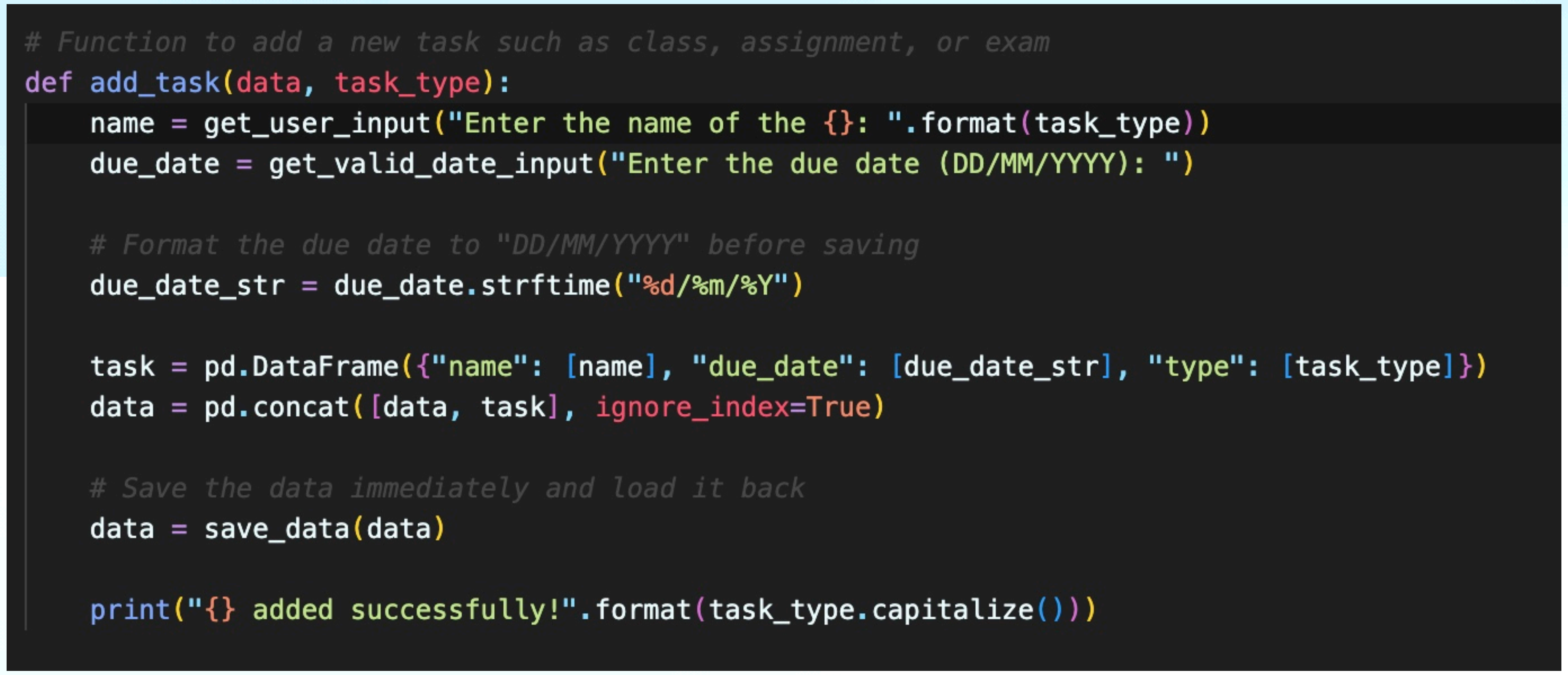
## Save Data



```python
# Function to save data to JSON file using pandas
def save_data(data):
    print("Data to be saved:")
    print(data)
    data.to_json("study_planner.json", orient="records", date_format="iso", indent=2)
```

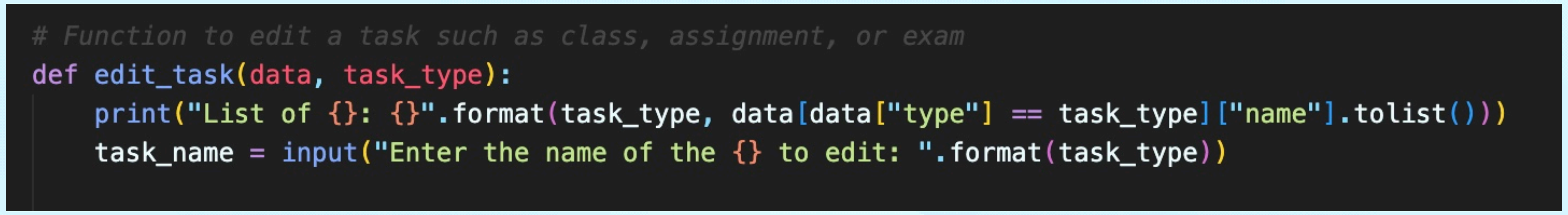
# Code Overview

## Add Task



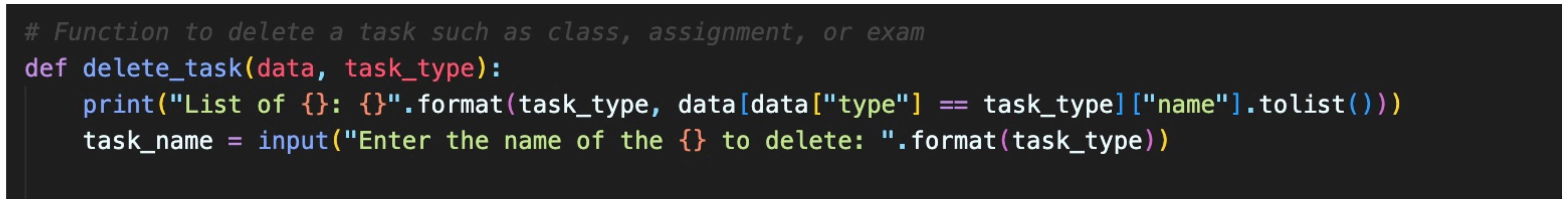
```python
# Function to add a new task such as class, assignment, or exam
def add_task(data, task_type):
    name = get_user_input("Enter the name of the {}: ".format(task_type))
    due_date = get_valid_date_input("Enter the due date (DD/MM/YYYY): ")

    # Format the due date to "DD/MM/YYYY" before saving
    due_date_str = due_date.strftime("%d/%m/%Y")

    task = pd.DataFrame({"name": [name], "due_date": [due_date_str], "type": [task_type]})
    data = pd.concat([data, task], ignore_index=True)

    # Save the data immediately and load it back
    data = save_data(data)

    print("{} added successfully!".format(task_type.capitalize()))
```

## Edit Task



```python
# Function to edit a task such as class, assignment, or exam
def edit_task(data, task_type):
    print("List of {}: {}".format(task_type, data[data["type"] == task_type]["name"].tolist()))
    task_name = input("Enter the name of the {} to edit: ".format(task_type))
```

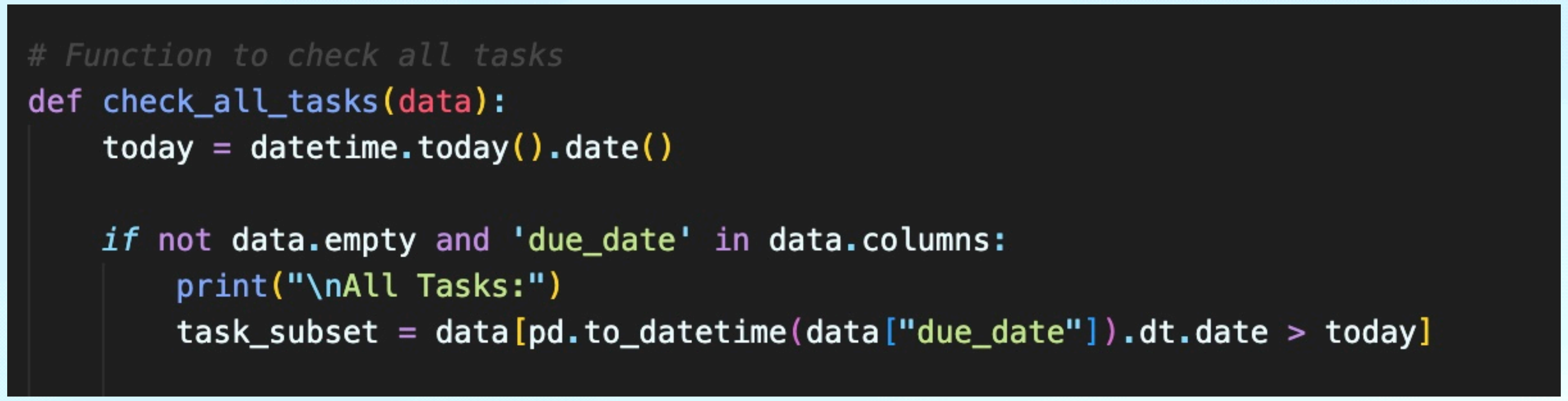
## Delete Task



```python
# Function to delete a task such as class, assignment, or exam
def delete_task(data, task_type):
    print("List of {}: {}".format(task_type, data[data["type"] == task_type]["name"].tolist()))
    task_name = input("Enter the name of the {} to delete: ".format(task_type))
```

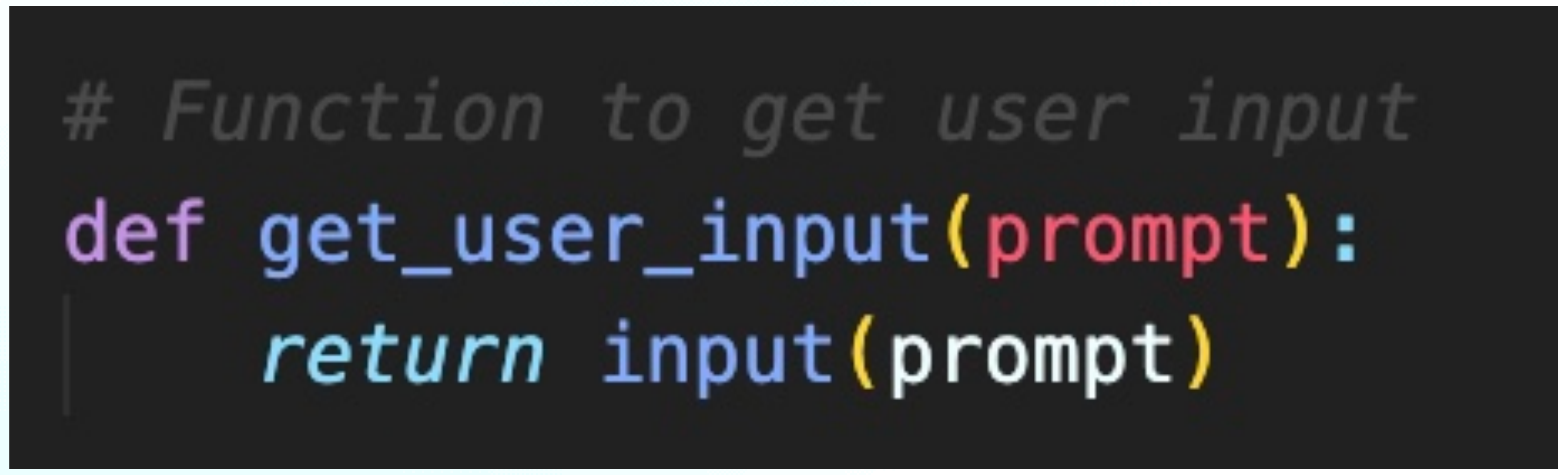
# Code Overview

## Check Task



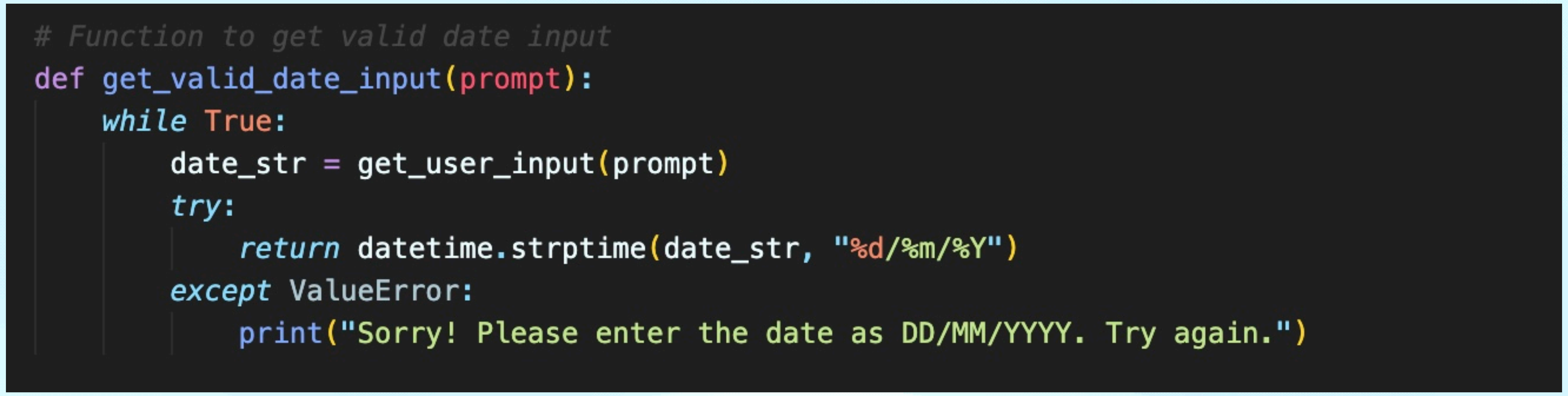
```python
# Function to check all tasks
def check_all_tasks(data):
    today = datetime.today().date()

    if not data.empty and 'due_date' in data.columns:
        print("\nAll Tasks:")
        task_subset = data[pd.to_datetime(data["due_date"]).dt.date > today]
```

## Get User Input



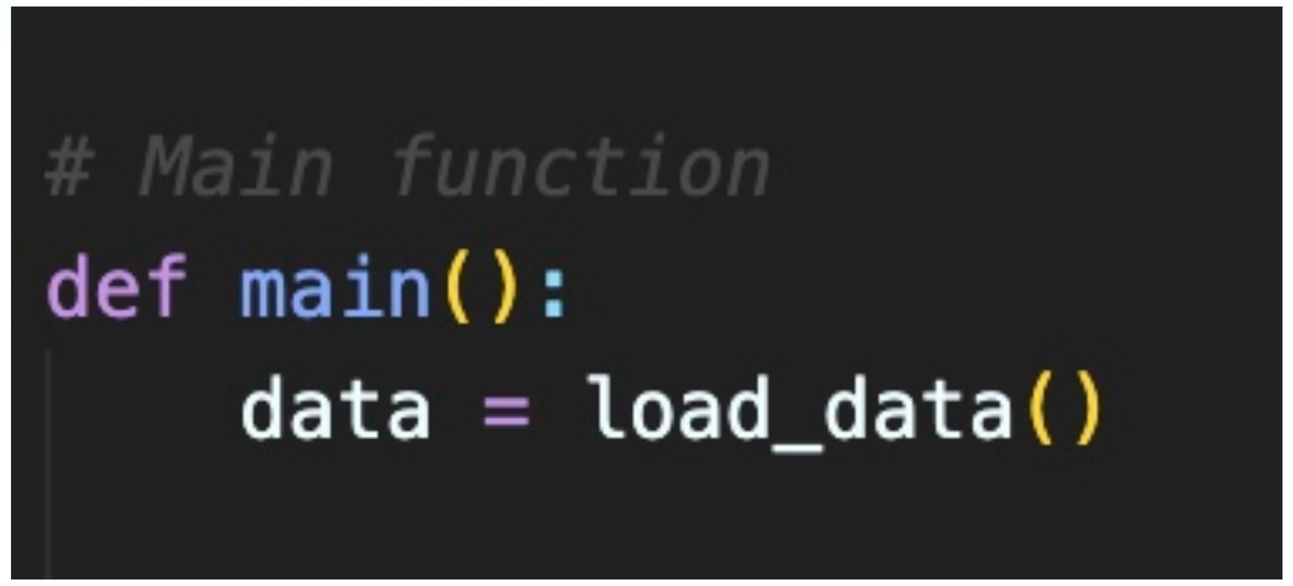
```python
# Function to get user input
def get_user_input(prompt):
    return input(prompt)
```

## Get Valid Date



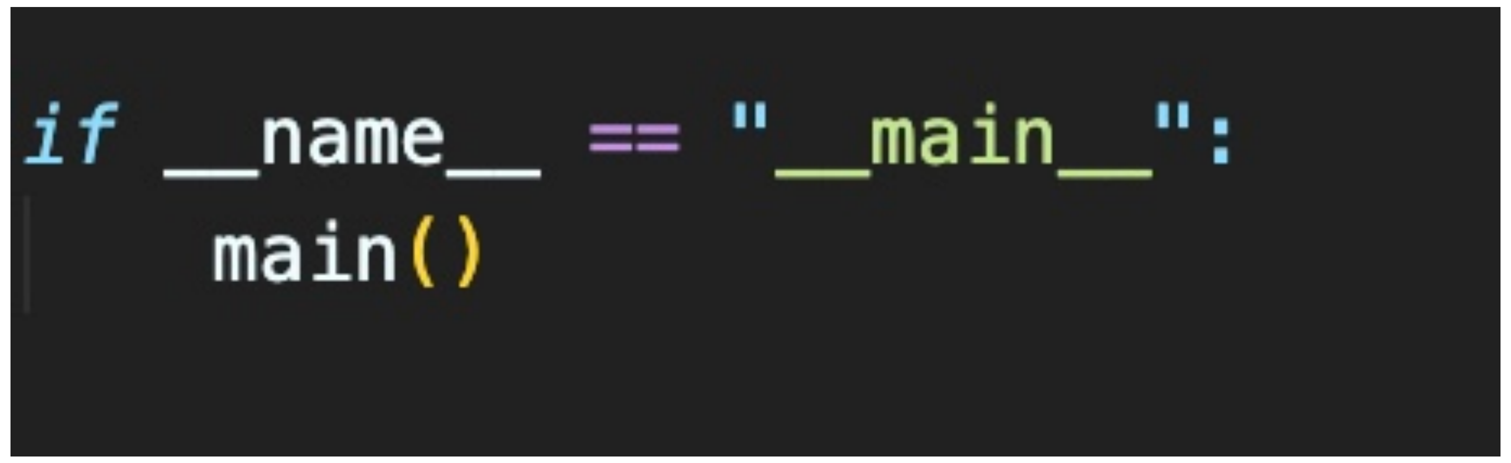
```python
# Function to get valid date input
def get_valid_date_input(prompt):
    while True:
        date_str = get_user_input(prompt)
        try:
            return datetime.strptime(date_str, "%d/%m/%Y")
        except ValueError:
            print("Sorry! Please enter the date as DD/MM/YYYY. Try again.")
```

## Main



```python
# Main function
def main():
    data = load_data()
```

## Main Execution block



```python
if __name__ == "__main__":
    main()
```

## Review of the Development/Build Process for the Study Planner Application

### Development and Build Process

The development and build process for the study planner application was a rewarding journey, which combining creative problem-solving with practical software development skills. The process' progressive nature made it possible for features to be continuously improved and refined.

### Challenges

One notable challenge was handling date input from users and ensuring robust error handling. Implementing a reliable mechanism to validate date formats and gracefully handle user mistakes required careful consideration. Additionally, integrating external packages and managing dependencies introduced some complexities, especially concerning version compatibility.

## Review of the Development/Build Process for the Study Planner Application

### Ethical Considerations

Ethical considerations were prioritised throughout the development process. Privacy and data security were considered, leading to the decision to store task data locally and not involve external servers to maintain ethical standards.

### Favourite Parts

One of my favorite aspects of the development process was implementing the feature to check all tasks. This feature added a practical dimension to the application, allowing users to get a quick overview of upcoming tasks. The use of coloured text in the command-line interface also added a visually appealing touch, enhancing the overall user experience.

# Thank you

## Study Planner Terminal Application

**Zac_Xenitopoulos_T1A3**