بسم الله رحمان رحیم

شرح پی دی اف پروژه :

۱ – کل کد پروژه

۲ ـ ساختمان داده های استفاده شده

۳ ـ بخش هایی که توسط هر دانشجو زده شده

Source code :

Note: If you have problem to display the code, you can use following link in my git hub page

Link :

```
1   class CellType:
2       def __init__(self, element='Header', next=None):
3           self.element = element
4           self.next = next
5
6       def print_cell(self):
7           print(self.element, end=",")
8
9
10  class SList:
11      def __init__(self):
12          self.header = CellType()
13          self.size = 0
14
15      def end_list(self):
16          h = self.header
17          while h.next is not None:
18              h = h.next
19          return h
20
21      def append(self, element):
22          endl = self.end_list()
23          cellnode = CellType(element)
24          endl.next = cellnode
25          self.size += 1
26
27      def __iter__(self):
28          current = self.header.next
29          while current :
30              yield current.element
31              current = current.next
32      def pop(self, value):
33          element = self.header.next
34          previous = self.header
35          while element :
36              if element.element == value:
37                  previous.next = element.next
38                  self.size -= 1
39                  return
40              previous = element
41              element = element.next
42
43  class Time:
44      def __init__(self, hour, minute):
45          self.hour = hour
46          self.minute = minute
47
48      def show(self):
```

```
        return f"{self.hour:02}  :  {self.minute:02}"


class Lessons:
    def __init__(self, name, location, lesson_id, next_lesson=None,
time_lesson=(0, 0)):
        self.name = name
        self.location = location
        self.lesson_id = lesson_id
        self.time_lesson = time_lesson
        self.next_lesson = next_lesson

    def print_lessons(self):
        current = self
        while current :
            hour, minute = current.time_lesson
            print(f"lesson: {current.name}
(ID:{current.lesson_id})\tlocation:
{current.location}\tTime:{hour:02}:{minute:02}")
            current = current.next_lesson

    def add_lesson(self, name, location, lesson_id):
        new_lesson = Lessons(name, location, lesson_id)
        if not self:
            return new_lesson
        current = self
        while current.next_lesson:
            current = current.next_lesson
        current.next_lesson = new_lesson

    def remove_lesson1(self, lesson_id):
        current = self
        if current.lesson_id == lesson_id:
            return current.next_lesson
        while current.next_lesson :
            if current.next_lesson.lesson_id == lesson_id:
                current.next_lesson = current.next_lesson.next_lesson
                return self
            current = current.next_lesson
        return self


def find_students_by_lesson(lesson_id, student_list):
    print(f"!!!!Searching for students who took lesson with ID
{lesson_id}!!!!")

    matched_students = []

    for student in student_list:
        for pointer in student.P_profile.week_lessons:
            lesson = pointer.lesson
            while lesson:
                if lesson.lesson_id == lesson_id:
                    matched_students.append((student, lesson.location))
                    break
                lesson = lesson.next_lesson
            if matched_students and matched_students[-1][0] == student:
                break
```

```
107
108     if matched_students:
109         for student, location in matched_students:
110             print(student.P_profile.cout(), student.show(), f"Class
111 Location: {location}")
112     else:
113         print("No students found for this lesson ID.")
114
115
116 class Pointer:
117     def __init__(self, day, lesson=None):
118         self.day = day
119         self.lesson = lesson
120
121     def print_pointer(self):
122         print(f"{self.day}: ")
123         if self.lesson:
124             self.lesson.print_lessons()
125         else:
126             print("No lesson")
127
128
129 class Profile:
130     def __init__(self, fname, lname, age, week_lessons=None):
131         self.fname = fname
132         self.lname = lname
133         self.age = age
134         self.week_lessons = week_lessons if week_lessons else []
135
136     def cout(self):
137         return f"Name: {self.fname} {self.lname}, Age: {self.age}"
138
139     def show_week_lessons(self):
140         print(f"Weekly lessons for {self.fname} {self.lname}:")
141         for pointer in self.week_lessons:
142             pointer.print_pointer()
143
144     def add_lesson(self, day, lesson):
145         for pointer in self.week_lessons:
146             current = pointer.lesson
147             while current :
148                 if current.name== lesson.name:
149                     print(f"Error: {self.fname} already has the class
150 '{lesson.name}' on another day.")
151                     return
152                 current = current.next_lesson
153
154         for pointer in self.week_lessons:
155             if pointer.day == day:
156                 current = pointer.lesson
157                 while current:
158                     if current.time_lesson == lesson.time_lesson:
159                         print(f"Error: {self.fname} already has a class
160 at this time on {day}.")
161                         return
162                     elif current.lesson_id == lesson.lesson_id:
163                         print(f"Error: {self.fname} already has this
164 class.")
```

```
                        return
                    current = current.next_lesson

                if pointer.lesson is None:
                    pointer.lesson = lesson
                else:
                    current = pointer.lesson
                    while current.next_lesson:
                        current = current.next_lesson
                    current.next_lesson = lesson

                print(f"Lesson '{lesson.name}' added on {day} at
{lesson.time_lesson}.")
                return
        print(f"Error: {day} is not in {self.fname}'s weekly
schedule.")

    def remove_lesson2(self, day, lesson_id):
        day_pointer = next((p for p in self.week_lessons if p.day ==
day), None)
        if not day_pointer or not day_pointer.lesson:
            print("Lesson not found or day has no lessons.")
            return

        current = day_pointer.lesson
        previous = None

        while current:
            if current.lesson_id == lesson_id:
                if previous is None:
                    day_pointer.lesson = current.next_lesson
                else:
                    previous.next_lesson = current.next_lesson
                print(f"Lesson '{current.name}' removed successfully.")
                return
            previous = current
            current = current.next_lesson

        print("Lesson ID not found.")


class Student:
    def __init__(self, student_id, next_s=None, p_profile=None):
        self.ID = student_id
        self.next_s = next_s
        self.P_profile = p_profile

    def show(self):
        return f"Student ID: {self.ID}"


l1 = Lessons("Math", 1205, "MATH_SATURDAY")
l2 = Lessons("Physic", 1314, "PHYSIC_SATURDAY",l1)
l3 = Lessons("English", 1134, "ENGLISH_SUNDAY")
l4 = Lessons("Data Structure", 1502, "DATA STRUCTURE_SUNDAY",l3)
l5 = Lessons("Algorithm", 1414, "ALGORITHM_MONDAY")
l6 = Lessons("Programming", 1512, "PROGRAMMING_MONDAY",l5)
l7 = Lessons("Culture Of Iran", 2212, "CULTURE OF IRAN_TUESDAY")
```

```
l8 = Lessons("Islamic History", 1115, "ISLAMIC HISTORY_TUESDAY",l7)
l9 = Lessons("Logic System", 1513, "LOGIC SYSTEM_WEDNESDAY")
l10 = Lessons("Electric System", 1234, "ELECTRIC SYSTEM_WEDNESDAY",l9)
l11 = Lessons("Web Design", 1234, "WEB DESIGN_THURSDAY")

m = [
    Pointer("Saturday", l2),
    Pointer("Sunday", l4),
    Pointer("Monday", l6),
    Pointer("Tuesday", l8),
    Pointer("Wednesday", l10),
    Pointer("Thursday", l11),
    Pointer("Friday")
]
n = [
    Pointer("Saturday"),
    Pointer("Sunday"),
    Pointer("Monday"),
    Pointer("Tuesday"),
    Pointer("Wednesday"),
    Pointer("Thursday"),
    Pointer("Friday")
]
P1 = Profile("Farhan", "Golestani", 20, m)
S1 = Student("1643490", None, P1)
P2 = Profile("Amirali", "Sadeghi", 22,n)
S2 = Student("1658438", S1, P2)
Student_List = [S1, S2]
a = SList()
a.append(S1)
a.append(S2)


# new_lesson = Lessons("math", 1205, "MATCH_SUNDAY", time_lesson=10)
# P1.add_lesson("Sunday", new_lesson)
#
# conflicting_lesson = Lessons("history", 1302, "HISTORY_SATURDAY",
time_lesson=10)
# P1.add_lesson("Saturday", conflicting_lesson)
#
# P1.remove_lesson("PHYSIC_SATURDAY")


def f(choice):
    match choice:
        case "1":
            student_id = input("Enter student ID to view schedule: ")
            student = next((s for s in a if s.ID == student_id), None)
            if student:
                student.P_profile.show_week_lessons()
            else:
                print("Student not found.")
            main()
        case "2":
            student_id = input("Enter student ID to add a lesson: ")
            student = next((s for s in a if s.ID == student_id), None)
            if student:

```

```
                day = input("Enter day to add the lesson (e.g.,
'Monday'): ")
                name = input("Enter name of the lesson: ")
                location = input("Enter class of lesson: ")
                lesson_id = input("Enter lesson ID: ").upper()
                time_input = input("Enter lesson time (HH:MM format,
24-hour): ")
                try:
                    hour, minute = map(int, time_input.split(":"))
                    new_lesson = Lessons(name, location, lesson_id,
time_lesson=(hour, minute))
                    student.P_profile.add_lesson(day, new_lesson)
                except ValueError:
                    print("Invalid time format. Please enter the time
as HH:MM.")

            else:
                print("Student not found.")
            main()
        case "3":
            student_id = input("Enter student ID to remove a lesson: ")
            student = next((s for s in a if s.ID == student_id), None)
            if student:
                day = input("Enter the day of the lesson to remove
(e.g., 'Monday'): ")
                lesson_id = input("Enter lesson ID to remove: ")
                student.P_profile.remove_lesson2(day, lesson_id)
            else:
                print("Student not found.")
            main()
        case "4":
            course = input("Enter name of course : ").upper()
            day = input("Enter day of course : ").upper()
            result = course + "_" + day
            find_students_by_lesson(result, a)
            main()
        case "5":
            l = [
                Pointer("Saturday"),
                Pointer("Sunday"),
                Pointer("Monday"),
                Pointer("Tuesday"),
                Pointer("Wednesday"),
                Pointer("Thursday"),
                Pointer("Friday")
                ]
            student_id=input("Enter student's ID : ")
            f_name=input("Enter student's name : ")
            l_name = input("Enter student's last name : ")
            age = int(input("Enter student's Age : "))
            new_profile = Profile(f_name , l_name, age , l)
            new_student = Student(student_id,None,new_profile)
            a.append(new_student)
            print("Student adding was successful.")
            main()
        case "6" :
            student_id = input("Enter student ID to remove : ")
            student= next((s for s in a if s.ID == student_id), None)
```

```
            if student:
                a.pop(student)
                print("Removing student was successfully")
            else:
                print("Student not found.")
            main()
        case "0":
            exit()
        case _:
            print("Invalid Choice! Try Again\n")
            main()


def main():
    print(
        "\n\t--- Main Menu ---\n\n"
        "\t1. View Student Schedule\n"
        "\t2. Add a Lesson\n"
        "\t3. Remove a Lesson\n"
        "\t4. Search a Lesson\n"
        "\t5. Add a student\n"
        "\t6. Remove a student\n"
        "\t0. Exit")
    choice = input("\nEnter your choice : ")
    f(choice)



# for i in a:
#     print(i.P_profile.cout(), i.show())
#     i.P_profile.show_week_lessons()

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

ساختمان داده های استفاده شده

:

۱ـ لیست پیوندی
کلاس هاCell Type وSlistاز روش لیست پیوندی پیروی میکنن
همچنینa یک لیستی از دانشجویان هست که به همین روش پیاده سازی شده
اشاره گر ۲ـ
کلاسpointer شامل لیستی از روزهای هفته هست که
ما میتونیم به درس های هریک از روزهای هفته دسترسی داشته باشیم
۳ـ درس ها
کلاسlessons بصورت لیست پیوندی درست شده که
میتونه هر درس رو به درس بعدی وصل کنه
۴ـ دانشجو و پروفایل
کلاس دانشجو دارای آیدی و اشاره گری به دانشجوی بعدی هست و
یک اشاره گر از نوع پروفایل که به پروفایل دانشجو اشاره میکنه
کلاس پروفایل هم که دارای اطلاعات دانشجو و
یک اشاره گر هست که به روزای هفته اشاره میکند

فعالیت هر دانشجو:

پوریا مردانشاهی :
بطور کلی نوشتن جزئیات (توابع – شرط ها - حلقه و...)
که در مورد ۵(اضافه کردن دانشجو) و مورد۶(حذف دانشجو) با ایشون بوده

پدرام فرجامی راد :
بطور کلی نوشتن جزئیات (توابع – شرط ها - حلقه و...)
که در مورد۲(اضافه کردن درس) و مورد۴(جستجو یک درس برای نمایش دادن دانشجوهایی که اون درس رو دارن) با ایشون بوده

فرهان گلستانی :
بطور کلی نوشتن تمامی کلاس های استفاده شده در پروژه وجزئیات(توابع – شرط ها - حلقه و...)
که در مورد ۳ (حذف کردن یک درس) مورد ۱(دیدن جزئیات دانشجو که شامل درس های هفتگی و اطلاعات شخصی میشه) با ایشون بوده

شکل ساختمان داده :