

پاسخ سوال برنامه نویسی

دوره استادی پایتون درسمن

پایتون پیشرفته



پاسخ تمرین شماره ۱:

در این راه حل به جای ۱۰ نمونه از کلاس دانش آموز، تنها ۵ نمونه تعریف شده است، با این هدف که تنها الگویی برای حل این مسأله است.

```

1  ### class for student in school
2  # =====
3  class Student:
4      school_Name = "SABA School" # class variable
5
6      def __init__(self, student_Code, name, family, level): # instance method
7          self.student_Code = student_Code
8          self.name = name
9          self.family = family
10         self.level = level
11
12     # this function shows list of courses for each level
13     @staticmethod
14     def show_Courses_By_Level(level):
15         if level == 1:
16             courses_List1=["C1","C2","C3","C4","C5"] # عنوان دروسهای پایه اول ابتدایی
17             for course in courses_List1:
18                 print(course,end=" ")
19         elif level == 2:
20             courses_List2=["C1","C2","C3","C4","C5","C6","C7","C8"] # عنوان دروسهای پایه دوم ابتدایی
21             for course in courses_List2:
22                 print(course,end=" ")
23         elif level == 3:
24             courses_List2=["C1","C2","C3","C4","C5","C6","C7","C8","C9"] # عنوان دروسهای پایه سوم ابتدایی
25             for course in courses_List2:
26                 print(course,end=" ")
27         elif level == 4:
28             courses_List2=["C1","C2","C3","C4","C5","C6","C7","C8","C9","C10"] # عنوان دروسهای پایه چهارم ابتدایی
29             for course in courses_List2:
30                 print(course,end=" ")
31         elif level == 5:
32             courses_List2=["C1","C2","C3","C4","C5","C6","C7","C8","C9"] # عنوان دروسهای پایه پنجم ابتدایی
33             for course in courses_List2:
34                 print(course,end=" ")
35         elif level == 6:
36             courses_List2=["C1","C2","C3","C4","C5","C6","C7","C8","C9","C10","C11"] # عنوان دروسهای پایه ششم ابتدایی
37             for course in courses_List2:
38                 print(course,end=" ")
39
40     # this function changes value of class variable
41     @classmethod
42     def change_School_Name (Student,new_Name):
43         Student.school_Name = new_Name
44         return Student.school_Name
45
46     # this function creates a hash code for each object
47     def __hash__(self): # instance method
48         return hash(self.student_Code)+ hash(self.name) + hash(self.family) + hash(self.level)
49
50     # this function compares the instances of the class
51     def __eq__(self,obj2): # instance method
52         if not isinstance(obj2,Student):
53             return False
54         return self.student_Code == obj2.student_Code and self.name ==obj2.name and self.family == obj2.famil
55
56     def __str__(self): # instance method
57         return f"Code:{self.student_Code}\tName:{self.name}\tFamily:{self.family}\tLevel:{self.level}"
58 # =====
59 #----- main program -----
60 students_Set = set()
61 for i in range(5):
62     code = int(input("Enter Student Code: "))
63     name = input("Enter Student Name: ")
64     family = input("Enter Student family: ")
65     level = int(input("Enter Student Level: "))
66     student = Student(code, name, family, level)
67     students_Set.add(student)

```



```

68 | print("-----")
69
70 print("List of Students:\n")
71 for student in students_Set:
72 | print(student)
73 print("-----")
74 #####
75 level = input("Enter Level: ")
76 print("-----")
77 print(f"List of Courses for Level({level}): \n")
78 Student.show_Courses_By_Level(3)
79 print()
80 print("-----")
81 #####
82 print("School Name changed to", Student.change_School_Name ("Narges"), "School")
83 print("-----")
84 # ===== output =====
85 #-----
Enter Student Code: 1
Enter Student Name: Sara
Enter Student family: Salimi
Enter Student Level: 3
-----
Enter Student Code: 2
Enter Student Name: Sina
Enter Student family: Sadr
Enter Student Level: 5
-----
Enter Student Code: 3
Enter Student Name: Bahar
Enter Student family: Karami
Enter Student Level: 6
-----
Enter Student Code: 1
Enter Student Name: Sara
Enter Student family: Salimi
Enter Student Level: 3
-----
Enter Student Code: 4
Enter Student Name: Maryam
Enter Student family: Majd
Enter Student Level: 2
-----
List of Students:

Code:3 Name: Bahar Family: Karami Level: 6
Code:4 Name: Maryam Family: Majd Level: 2
Code:2 Name: Sina Family: Sadr Level: 5
Code:1 Name: Sara Family: Salimi Level: 3
-----
Enter Level: 5
-----
List of Courses for Level(5):

C1 C2 C3 C4 C5 C6 C7 C8 C9
-----
School Name changed to Narges School
-----

```



پاسخ تمرین شماره ۲:

```

1  ### class for participants
2  # =====
3  from abc import ABC, abstractmethod
4  # =====
5  #----- classes and subclasses -----
6  ## parent class
7  class Participants(ABC):
8      def __init__(self, name, family, id_Number, major, address):
9          self.name = name
10         self.family = family
11         self.id_Number = id_Number
12         self.major = major
13         self.address = address
14
15         @abstractmethod
16         def calculation_Score(self):
17             pass
18
19         def _show_Participant_Information(self):
20             return f'Name:{self.name}\tFamily:{self.family}\tID Number:{self.id_Number}\tMajor:{self.major}\tAddress:{self.address}'
21
22  # -----
23  ## child class
24  class Participant_Free(Participants): # شرکت کننده آزاد
25      def __init__(self, pf_Code, name, family, id_Number, major, address, test_Score1, test_Score2):
26          Participants.__init__(self, name, family, id_Number, major, address)
27          self.__pf_Code = pf_Code
28          self.__test_Score1 = test_Score1 # نمره آزمون کتبی
29          self.__test_Score2 = test_Score2 # نمره مصاحبه
30
31          def calculation_Score(self):
32              final_Score = ( self.__test_Score1 + self.__test_Score2)/2
33              return final_Score
34
35          @property
36          def test_Score1(self):
37              return self.__test_Score1
38
39          @test_Score1.setter
40          def test_Score1(self, new_Score):
41              self.__test_Score1 = new_Score
42
43          @property
44          def test_Score2(self):
45              return self.__test_Score2
46
47          @test_Score2.setter
48          def test_Score2(self, new_Score):
49              self.__test_Score2 = new_Score
50
51          def __str__(self):
52              return f'Code:{self.__pf_Code}\t{self._show_Participant_Information()}\tScore:{self.calculation_Score()}'
53
54  # -----
55  ## child class
56  class Participants_Special(Participants): # دانش آموخته نخبه
57      def __init__(self, ps_Code, name, family, id_Number, major, address, university_Rank, grade_Point_Average):
58          Participants.__init__(self, name, family, id_Number, major, address)
59          self.__ps_Code = ps_Code
60          self.__university_Rank = university_Rank
61          self.__grade_Point_Average = grade_Point_Average

```



```

63 def calculation_Score(self):
64     if self.__university_Rank == 1 and self.__grade_Point_Average >= 18.5:
65         return (100+100)/2
66
67     elif self.__university_Rank == 1 and 17.5 <= self.__grade_Point_Average < 18.5:
68         return (100+80)/2
69
70     elif self.__university_Rank == 1 and 16 < self.__grade_Point_Average < 17.5:
71         return (100+60)/2
72
73     elif self.__university_Rank == 2 and self.__grade_Point_Average >= 18.5:
74         return (80+100)/2
75
76     elif self.__university_Rank == 2 and 17.5 <= self.__grade_Point_Average < 18.5:
77         return (80+80)/2
78
79     elif self.__university_Rank == 2 and 16 < self.__grade_Point_Average < 17.5:
80         return (80+60)/2
81
82     elif self.__university_Rank == 3 and self.__grade_Point_Average >= 18.5:
83         return (60+100)/2
84
85     elif self.__university_Rank == 3 and 17.5 <= self.__grade_Point_Average < 18.5:
86         return (60+80)/2
87
88     elif self.__university_Rank == 3 and 16 < self.__grade_Point_Average < 18.5:
89         return (60+60)/2
90
91 @property
92 def university_Rank(self):
93     return self.__university_Rank
94
95 @university_Rank.setter
96 def university_Rank(self, new_Rank):
97     self.__university_Rank = new_Rank
98
99 @property
100 def grade_Point_Average(self):
101     return self.__grade_Point_Average
102
103 @grade_Point_Average.setter
104 def grade_Point_Average(self, new_Average):
105     self.__grade_Point_Average = new_Average
106
107 def __str__(self):
108     return f"Code:{self.__ps_Code}\t{self._show_Participant_Information()}\tScore:{self.
        calculation_Score()}"
109
110 # -----
111 ## child class
112 class Employee(Participants): # کارمند قراردادی
113     def __init__(self, e_Code, name, family, id_Number, major, address, performance_Score,
        working_Years):
114         Participants.__init__(self, name, family, id_Number, major, address)
115         self.__e_Code = e_Code
116         self.__performance_Score = performance_Score
117         self.__working_Years = working_Years
118
119     def calculation_Score(self):
120         if 1 < self.__working_Years <= 5:
121             final_Score = self.__performance_Score + (self.__performance_Score * 0.1)
122             return final_Score
123         else:
124             final_Score = self.__performance_Score + (self.__performance_Score * 0.2)
125             return final_Score

```



```

126     @property
127     def performance_Score(self):
128         return self.__performance_Score
129
130     @performance_Score.setter
131     def performance_Score(self, new_Score):
132         self.__performance_Score = new_Score
133
134     @property
135     def working_Years(self):
136         return self.__working_Years
137
138     @working_Years.setter
139     def working_Years(self, new_Number):
140         self.__working_Years = new_Number
141
142     def __str__(self):
143         return f"Code:{self.__e_Code}\t{self._show_Participant_Information()}\tScore:{self.
            calculation_Score()}"
144
145 # ===== main program =====
146 participants_List = []
147 accepted_Participants_List = []
148 #####
149 # the instances of the Participant_Free class
150
151 for i in range(3):
152     code = input("Enter Code: ")
153     name = input("Enter Name: ")
154     family = input("Enter Family: ")
155     id_Number = input("Enter Id Number: ")
156     major = input("Enter Major: ")
157     address = input("Enter Address: ")
158     score1 = int(input("Enter Test Score1: "))
159     score2 = int(input("Enter Test Score2: "))
160     pf = Participant_Free(code, name, family, id_Number, major, address, score1, score2)
161     participants_List.append(pf)
162     if pf.calculation_Score() >= 90:
163         accepted_Participants_List.append(pf)
164     print("-----")
165 #####
166 # the instances of the Participants_Special class
167
168 for i in range(3):
169     code = input("Enter Code: ")
170     name = input("Enter Name: ")
171     family = input("Enter Family: ")
172     id_Number = input("Enter Id Number: ")
173     major = input("Enter Major: ")
174     address = input("Enter Address: ")
175     university = int(input("Enter University Rank: "))
176     average = float(input("Enter GPA: "))
177     ps = Participants_Special(code, name, family, id_Number, major, address, university, average)
178     participants_List.append(ps)
179     if ps.calculation_Score() >= 90:
180         accepted_Participants_List.append(ps)
181     print("-----")
182 #####
183 # the instances of the Employee class
184
185 for i in range(3):
186     code = input("Enter Code: ")
187     name = input("Enter Name: ")
188     family = input("Enter Family: ")
189     id_Number = input("Enter Id Number: ")
190     major = input("Enter Major: ")
191     address = input("Enter Address: ")
192     performance = int(input("Enter Performance Score: "))
193     working_years = float(input("Enter Working Years: "))
194     e = Employee(code, name, family, id_Number, major, address, performance, working_years)
195     participants_List.append(e)
196     if e.calculation_Score() >= 90:
197         accepted_Participants_List.append(e)
198     print("-----")
199 #####
200
201
202
203
204
205

```



```
206
207 print("List of Participants:\n")
208 for participant in participants_List:
209     print(participant)
210 print(100*"x")
211 print("List of Accepted Participants:\n")
212 for participant in accepted_Participants_List:
213     print(participant)
214 # =====
215 #----- output -----
```

```
Enter Code: pf-1
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Test Score1: 80
Enter Test Score2: 97
```

```
Enter Code: pf-2
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Test Score1: 70
Enter Test Score2: 80
```

```
Enter Code: pf-3
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Test Score1: 85
Enter Test Score2: 90
```

```
Enter Code: ps-1
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter University Rank: 3
Enter GPA: 18.5
```

```
Enter Code: ps-2
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter University Rank: 1
Enter GPA: 19
```

```
Enter Code: ps-3
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter University Rank: 2
Enter GPA: 19
```

```
Enter Code: ee-1
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Performance Score: 88
Enter Working Years: 5
```

```
Enter Code: ee-2
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Performance Score: 90
Enter Working Years: 3
```

```
Enter Code: ee-3
Enter Name: aa
Enter Family: bb
Enter Id Number: 000
Enter Major: cc
Enter Address: dd
Enter Performance Score: 70
Enter Working Years: 6
```

List of Participants:

Code:pf-1	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:88.5
Code:pf-2	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:75.0
Code:pf-3	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:87.5
Code:ps-1	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:80.0
Code:ps-2	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:100.0
Code:ps-3	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:90.0
Code:ee-1	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:96.8
Code:ee-2	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:99.0
Code:ee-3	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:84.0

List of Accepted Participants:

Code:ps-2	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:100.0
Code:ps-3	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:90.0
Code:ee-1	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:96.8
Code:ee-2	Name:aa	Family:bb	ID Number:000	Major:cc	Address:dd	Score:99.0

