**ლექცია 6 – ლექსიკონები dict**

**მონაცემების შენახვა, ჰეშირება**

ჩამოვთვალოთ რა ტიპის მონაცემები გავიარეთ: int, float, bool, str, list, dict.

[15, 23, 78, 32, "text", 65]   ===>  სია -- list

{'name': 'python'}             ===>  ლექსიკონი -- dict

სიის ელემენტზე წვდომა ხდება ინდექსის საშუალებით:

my\_list = [12, 58, 23, "John", [12, 58, 2.56]]

print(my\_list[3])

John

ლექსიკონის შემთხვევაში კი ელემენტს ვწვდებით გასაღების (key) საშუალებით:

my\_dict = {'name': "Ana"}

dict\_name1 = my\_dict['name']

dict\_name2 = my\_dict.get("name")

# dict\_name3 = my\_dict['age']       # KeyError: 'age'

dict\_name4 = my\_dict.get("age")

dict\_name5 = my\_dict.get("age", "incorect key")

print(dict\_name1)

print(dict\_name2)

print(dict\_name4)

print(dict\_name5)

Ana

Ana

None

incorect key

# შევცვალოთ ლექსკონის სტრუქტურა… რატომ არის შეცდომა?

my\_dict = {

  "student": {"name": "Paata", "age": 51}

}

name1 = my\_dict['name'] # KeyError: 'name'

name2 = my\_dict.get('name')

KeyError: 'name'

my\_dict = {

  "student": {"name": "Paata", "age": 51}

}

dict\_name1 = my\_dict['student']

dict\_name2 = my\_dict.get('student')

dict\_name3 = my\_dict['student']['name']

dict\_name4 = my\_dict.get('student').get('name')

print(dict\_name1)

print(dict\_name2, "\n")

print(dict\_name3)

print(dict\_name4, "\n")

{'name': 'Paata', 'age': 51}

{'name': 'Paata', 'age': 51}

Paata

Paata

dict\_name4 = my\_dict.get('student').get('name1') # None

dict\_name4 = my\_dict.get('studen').get('name') # 'NoneType' object has no attribute 'get'

აქ my\_dict.get('studen') აბრუნებს None ტიპს და მასზე არანაირი არითმეტიკული თუ სხვა ოპერაციები არ შეიძლება. პირველმა get() მეთოდმა უნდა მომცეს ლექსიკონი, რათა მეორემაც სწორად იმუშაოს.

students = {

  "students": [

    {"id": 1, "name": "Paata", "age": 24},

    {"id": 2, "name": "Nika", "age": 20},

    {"id": 3, "name": "Lasha", "age": 21},

  ],

  "grades": [

    {"Paata": "B"},

    {"Nika": "A"},

    {"Lasha": "C"},

  ]

}

# როგორ ავიღო ნიკას შედეგი?

# nika\_grade = students.get('grades')

[{'Paata': 'B'}, {'Nika': 'A'}, {'Lasha': 'C'}]

# nika\_grade = students['grades']

[{'Paata': 'B'}, {'Nika': 'A'}, {'Lasha': 'C'}]

# nika\_grade = students.get('grades')[1]

{'Nika': 'A'}

# nika\_grade = students.get('grades')[1]['Nika']

A

nika\_grade = students.get('grades')[1].get('Nika')

print(nika\_grade)

A

ახლა დავბეჭდოთ ასეთი ფორმატით:

Student\_id: 3, name: Lasha, Grade: C

students = {

  "students": [

    {"id": 1, "name": "Paata", "age": 24},

    {"id": 2, "name": "Nika", "age": 20},

    {"id": 3, "name": "Lasha", "age": 21},

  ],

  "grades": [

    {"Paata": "B"},

    {"Nika": "A"},

    {"Lasha": "C"},

  ]

}

st\_id = students['students'][2]['id']

st\_name = students['students'][st\_id - 1]['name']

st\_grade = students["grades"][st\_id - 1][st\_name]

print(f"Student\_id: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

Student\_id: 3, Name: Lasha, Grade: C

# დავბეჭდოთ ყველა სტუდენტის მონაცემები

students = {

  "students": [

    {"id": 1, "name": "Paata", "age": 24},

    {"id": 2, "name": "Nika", "age": 20},

    {"id": 3, "name": "Lasha", "age": 21},

  ],

  "grades": [

    {"Paata": "B"},

    {"Nika": "A"},

    {"Lasha": "C"},

  ]

}

for student in students.get('students'):

  st\_id = student['id']

  st\_name = student.get('name')

  st\_grade = students['grades'][st\_id - 1].get(st\_name)

  print(f"ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

ID: 1, Name: Paata, Grade: B

ID: 2, Name: Nika, Grade: A

ID: 3, Name: Lasha, Grade: C

# დავბეჭდოთ ყველა სტუდენტის მონაცემები სხვადასხვა id-ებით

students = {

  "students": [

    {"id": 20, "name": "Paata", "age": 24},

    {"id": 100, "name": "Nika", "age": 20},

    {"id": 1232, "name": "Lasha", "age": 21},

  ],

  "grades": [

    {"Paata": "B"},

    {"Nika": "A"},

    {"Lasha": "C"},

    {"Archil": "C"},

    {"Alexandre": "C"},

  ]

}

for student in students.get('students'):

  st\_id = student['id']

  st\_name = student.get('name')

  st\_grade = students['grades'][st\_id - 1].get(st\_name) # IndexError: list index out of range

  print(f"ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

Traceback (most recent call last):

File "e:\PyApp\2024-2025\BPWSO-16-PM\ლექცია 6 – ლექსიკონები\main.py", line 145, in <module>

st\_grade = students['grades'][st\_id - 1].get(st\_name)

~~~~~~~~~~~~~~~~~~^^^^^^^^^^^

IndexError: list index out of range

grades სიაში st\_id არ ემთხვევა სიის ინდექსებს 0..1..2..   20, 100, 1232

ჩავწეროთ: st\_grade = '' და გავუშვათ პროგრამა...

ID: 20, Name: Paata, Grade:

ID: 100, Name: Nika, Grade:

ID: 1232, Name: Lasha, Grade:

როგორც ვხედავთ grade-ებს არ ბეჭდავს, რაც ბუნებრივია, ჩვენ st\_grade ცვლადს მივანიჭეთ ცარიელი სტრიქონი...

st\_grade ცვლადს უნდა მივანიჭოთ სტუდენტების სახელები student['name'].

students = {

  "students": [

    {"id": 20, "name": "Paata", "age": 24},

    {"id": 100, "name": "Nika", "age": 20},

    {"id": 1232, "name": "Lasha", "age": 21},

  ],

  "grades": [

    {"Nika": "A"},

    {"Paata": "B"},

    {"Lasha": "C"},

    {"Archil": "C"},

    {"Alexandre": "C"},

  ]

}

for student in students.get('students'):

  st\_id = student['id']

  st\_name = student.get('name')

  # st\_grade = students['grades'][st\_id - 1].get(st\_name)

  # st\_grade = ''

  # st\_grade = students.get('grades')

st\_grade = ''

  for grade in students.get('grades'):

    if st\_name in grade:

      st\_grade = grade.get(st\_name)

  print(f"ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

ID: 20, Name: Paata, Grade: B

ID: 100, Name: Nika, Grade: A

ID: 1232, Name: Lasha, Grade: C

ახლა, ასეთი ვარიანტი განვიხილოთ: grades-ებიდან ამოვიღოთ არჩილის შეფასების სტრიქონი, ხოლო students-ში არჩილის მონაცემები ჩავამატოთ. ციკლში კი – st\_grade = "Grade not found" და დავბეჭდოთ მონაცემები.

# დავბეჭდოთ ყველა სტუდენტის მონაცემები სხვადასხვა id-ებით

students = {

  "student": [

    {"id": 20, "name": "Paata", "age": 24},

    {"id": 100, "name": "Nika", "age": 20},

    {"id": 1232, "name": "Lasha", "age": 21},

**{'id': 8564, "name": "Archil", "age": 32},**

  ],

  "grades": [

    {"Paata": "B"},

    {"Nika": "A"},

    {"Lasha": "C"},

    {"Alexandre": "C"},

  ]

}

for student in students.get('student'):

  st\_id = student.get('id')

  st\_name = student.get('name')

**st\_grade = "Grade not found"**

  for grade in students.get('grades'):

    if st\_name in grade:

      st\_grade = grade.get(st\_name)

  print(f"Student ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

Student ID: 20, Name: Paata, Grade: B

Student ID: 100, Name: Nika, Grade: A

Student ID: 1232, Name: Lasha, Grade: C

Student ID: 8564, Name: Archil, Grade: Grade not found

თუ grades-ბში გვეწერება {"lasha": "C"}, მაშინ დაბეჭდავს ზუსტად არჩილისნაირად.

მონაცემებში შესაძლებელია სახელების დამთხვევა. თუ გვექნება ერთნაირ სახელიანი სტუდენტები, მაშინ მათი შეფასებებიც უნდა შევიტანოთ grades-ებში.

students = {

  "student": [

    {"id": 20, "name": "Paata", "age": 24},

**{"id": 25, "name": "Paata", "age": 21},**

**{"id": 31, "name": "Nika", "age": 22},**

    {"id": 100, "name": "Nika", "age": 20},

    {"id": 1232, "name": "Lasha", "age": 21},

    {'id': 8564, "name": "Archil", "age": 32},

  ],

  "grades": [

    {"Paata": "B"},

**{"Paata": "A"},**

    {"Nika": "A"},

**{"Nika": "B"},**

    {"Lasha": "C"},

    {"Archil": "A"},

  ]

}

for student in students.get('student'):

  st\_id = student.get('id')

  st\_name = student.get('name')

  st\_grade = "Grade not found"

  for grade in students.get('grades'):

    if st\_name in grade:

      st\_grade = grade.get(st\_name)

  print(f"Student ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

Student ID: 20, Name: Paata, Grade: A

Student ID: 25, Name: Paata, Grade: A

Student ID: 31, Name: Nika, Grade: B

Student ID: 100, Name: Nika, Grade: B

Student ID: 1232, Name: Lasha, Grade: C

Student ID: 8564, Name: Archil, Grade: A

როგორც ხედავთ ორივე პაატას და ნიკას grade-ები ერთნაირია, შესაბამისად – A და B. ჩაწერა შეფასების ბოლო მონაცემები.

ჩვენ სტუდენტები (students) და მათი შეფასებები (grades) ერთმანეთთან სახელებითაა დაკავშირებული. ამიტომაც შემოწმებისას (მეორე ციკლი) ვამოწმებთა სახელებს (სტუდენტის სახელი თუ არის შეფასების ლექსიკონში). დადებითი პასუხის შემთხვევაში შეფასებების ლექსიკონიდან შეფასების ბოლო მნიშვნლობა ენიჭება ცვლად st\_grade-ს. ეს რომ გამოვასწოროთ, საჭიროა სტიდენტების შესაბამისი ID-იბიც ჩავწეროთ grades-ში.

students = {

  "student": [

    {"id": 20, "name": "Paata", "age": 24},

    {"id": 25, "name": "Paata", "age": 21},

    {"id": 31, "name": "Nika", "age": 22},

    {"id": 100, "name": "Nika", "age": 20},

    {"id": 1232, "name": "Lasha", "age": 21},

    {'id': 8564, "name": "Archil", "age": 32},

  ],

  "grades": [

    {**"id": 20,** "Paata": "B"},

    {**"id": 25,** "Paata": "A"},

    {**"id": 31,** "Nika": "A"},

    {**"id": 100,** "Nika": "B"},

    {**"id": 1232,** "Lasha": "C"},

    {**"id": 8564,** "Archil": "A"},

  ]

}

for student in students.get('student'):

  st\_id = student.get('id')

  st\_name = student.get('name')

  st\_grade = "Grade not found"

  for grade in students.get('grades'):

    if st\_id == grade.get('id'):

      st\_grade = grade.get(st\_name)

  print(f"Student ID: {st\_id}, Name: {st\_name}, Grade: {st\_grade}")

Student ID: 20, Name: Paata, Grade: B

Student ID: 25, Name: Paata, Grade: A

Student ID: 31, Name: Nika, Grade: A

Student ID: 100, Name: Nika, Grade: B

Student ID: 1232, Name: Lasha, Grade: C

Student ID: 8564, Name: Archil, Grade: A

**ჰეშირება**

როგორ ინახავს ლექსიკონი გასაღებებს

dict1 = {"name": "Ana"}

print(dict1['name'])

dict1['name'] = 'John'

print(dict1['name'])

Ana

John

dict1 = {"name": "Ann", "name": "John"}

print(dict1['name'])

John

ამა ჰქვია ჰეშირება. პითონი ცხრილს აკეთებს და უნიკალური გასაღებებს ამ ცხრილის სვეტს ქმნის. ერთნაირი გასაღებების შემთხვევაში შექმნილ სვეტში გადააწერს ძველ მნიშვნელობას ახლით.

print(hash("name"))

print(hash("name"))

print(hash("names"))

dict1 = {"name": "Ann", "name": "John", "name": "Cat"}

print(dict1['name'])

-5861579566788208999

-5861579566788208999

8171414946176579168

Cat

dict1 = {

  'student': {'name': 'Paata'},

  'name': 'Nika',

  'grade': 'A'

}

print(dict1['student'])

print(dict1['student']['name'])

print(dict1['name'])

print(dict1['grade'])

{'name': 'Paata'}

Paata

Nika

A