2-зертхана: Python тіліндегі функциялар және өзгермейтіндік

Мақсат:

Python-да функционалдық бағдарламалау контекстінде функциялар және деректердің өзгермейтіндігі туралы түсініктерді зерттеу. Мақсаты студенттердің модульдік, тиімді және қауіпсіз кодты жасау үшін функцияларды қалай пайдалануға болатынын, сондай-ақ жанама әсерлердің алдын алу үшін деректердің өзгермейтіндігінің маңыздылығын түсінуді тереңдету.

Тапсырмалар:

1. Python тіліндегі функцияларды үйрену ерекшеліктері:

- Функциялар мен әдістердің айырмашылығын түсіну.

- Функциялар контекстінде жергілікті және ғаламдық айнымалыларды зерттеу.

- Ауыспалы ауқым ұғымын түсіну.

2. Деректердің өзгермейтіндігін түсіну және қолдану:

- Python тілінде қандай деректер түрлері өзгермейтінін білу.

- Функциялардың қауіпсіздігі мен тазалығына өзгермейтіндіктің әсерін түсіну.

- өзгермейтін деректерді пайдаланудың практикалық мысалдарын қарастыру.

3. Таза функцияларды құрастыру және пайдалану: - Функцияны «таза» ететін нәрсені түсіну.

- Таза функцияларды пайдаланудың артықшылықтарын зерттеу.

- таза функциялардың мысалдарын орындау және оларды таза емес функциялармен салыстыру.

4. Функциялармен және өзгермейтін деректер құрылымдарымен жұмыс істеу тәжірибесі: - Алған білімдерін практикалық мысалдар мен тапсырмаларда қолдану.

- функционалдық сипаттамаларын жақсарту үшін бар кодты талдау және рефакторинг.

5. Функционалдық стильдің кодтың өнімділігі мен оқылуына әсерін талқылау:

- Функционалдық стиль мен өнімділік арасындағы сәйкестіктерді талдау.

- Функционалдық стильде жазылған кодтың оқылу және қолдау мәселелерін қарастыру.

Жеке тапсырмалар:

Әрбір студентке топ тізіміндегі оның нөміріне сәйкес бірегей тапсырма беріледі (SSO қараңыз).Бұл тапсырмалар таза функцияларды жүзеге асыруға, жоғары ретті функцияларды жасауға және өзгермейтін деректер құрылымдарын пайдалануға бағытталған:

1. Валюта айырбастау функциясы

- Бастапқы деректерді өзгертпейтін валютаны айырбастауға арналған таза функцияны жазыңыз.

2. Функция генераторы

- Санды көрсетілген қуатқа көтеру үшін басқа функцияны қайтаратын жоғары ретті функцияны жасаңыз.

3. Қауіпсіз тізім

- Бастапқы тізім өзгеріссіз қалуы үшін әрекеттерді орындамас бұрын тізімді көшіретін функцияны орындаңыз.

4. Қорытындылау функциясы

- Кортеждегі барлық элементтерді қосу үшін таза функцияны жазыңыз.

5. Композиция функциясы

- Екі функцияны қабылдайтын және олардың құрамын қайтаратын жоғары ретті функцияны жасаңыз.

6. Сүзгі функциясы

- Түпнұсқаны өзгертпей, жаңа тізімді қайтарып, сандар тізімін сүзуге арналған таза функцияны іске қосыңыз.

7. Салық калькуляторы

- Түпнұсқа мәліметтерді өзгертпей, салықты табыс негізінде есептейтін функцияны жазыңыз.

8. Кері шақыру функциясы

- Кері шақыру функциясын қабылдайтын және оны тізім элементтеріне қолданатын жоғарырақ ретті функцияны жасаңыз.

9. Кездейсоқ сандар генераторы

- Ғаламдық күйді өзгертпей, кездейсоқ сандар тізімін тудыратын таза функцияны жазыңыз.

10. Есте сақтау функциясы

- Рекурсивті функцияларды есептеуді жылдамдату үшін есте сақтау функциясын орындаңыз.

11. Жолды бөлу функциясы

- Жолды қабылдайтын және бастапқы жолды өзгертпей сөздер тізімін қайтаратын таза функция жасаңыз.

12. Минималды және максималды іздеу функциясы

- Өзгермейтін кортеждегі ең кіші және ең үлкен сандарды табатын функцияны жазыңыз.

13. Деректерді түрлендіру функциясы

- Бастапқы тізімді өзгертпей, сөздіктер тізімін түрлендіретін функцияны іске асыру.

def f(data):  
 def f2(d):  
 dict = d.copy()  
 if 'val' in dict:  
 dict['val'] += 10  
 return dict  
 d2 = list(map(f2, data))  
 return d2  
d = [{'name': 'Umida', 'val': 1}, {'name': 'Damira', 'val': 20}]  
d2 = f(d)  
print("Original:", d)  
print("data - 2:", d2)

14. Сұрыптау функциясы

- Түпнұсқа тізімді өзгеріссіз қалдырып, тізімнің көшірмесін сұрыптау үшін таза функция жасаңыз.

15. Предикаттар функциясының генераторы

- Берілген шарт негізінде предикат функциясын жасайтын және қайтаратын жоғары ретті функцияны жазыңыз.

Бұл тапсырмалар Python-да функционалдық бағдарламалау контекстінде функцияның тазалығын, деректердің өзгермейтіндігін және жоғары ретті функцияларды құруды терең түсінуге ықпал етеді.

Бағалау критерийлері:

- Жеке есепті шешу үшін код жазу: 1 ұпай

- Қорғау кезінде жазылған кодты түсіндіру және түсіну: 2 ұпай - Мұғалім таңдаған теориялық сұрақтардың біріне жауап: 1 ұпай

Дайындық сұрақтары:

1. Функционалдық программалау контекстіндегі таза функция дегеніміз не?

- Мақсаты: Студенттердің таза функциялар түсінігін, оның ішінде жанама әсерлердің жоқтығын және енгізілген деректердің өзгермейтіндігін тексеру.

2. Деректердің өзгермейтіндігі функционалдық бағдарламалауға қалай әсер етеді?

- Мақсаты: Бағдарламалардың болжамдылығы мен сенімділігі үшін деректердің өзгермейтіндігінің маңыздылығын студенттің түсінуін бағалау.

3. Python тілінде жоғары ретті функцияға мысал келтіріңіз.

- Мақсаты: Оқушының жоғары ретті функциялар ұғымын қаншалықты түсінетінін және нақты мысал келтіре алатынын тексеру.

4. Python-да өзгермейтін деректер құрылымдарын қалай жүзеге асыра аламыз?

- Мақсаты: Студенттердің өзгермейтін деректер құрылымдарын құру және пайдалану жолын түсінуі.

5. Таза функцияларды қолдану қандай есептерді шешеді?

- Мақсаты: Студентке оңай тестілеу және жөндеу сияқты таза функцияларды пайдаланудың артықшылықтарын түсінуін қамтамасыз ету.

6. Таза функциялар бағдарлама жұмысын жақсарта ала ма? Егер солай болса, қалай?

- Мақсаты: Оқушылардың өнімділік контекстіндегі таза функциялардың әлеуетті артықшылықтарын түсінуін бағалау, мысалы, есте сақтау арқылы.

7. Бағдарламалаудағы жанама әсерлер қандай және олар таза функциялармен қалай байланысты?

- Мақсаты: Студенттің жанама әсерлер түсінігін және олардың таза функциялармен байланысын тексеру.

8. Жоғары ретті функция бағдарламалаудың ыңғайлылығы мен икемділігіне қалай әсер етуі мүмкін?

- Мақсаты: Модульдік және қайта пайдалануға болатын кодты жазуда студенттердің жоғары ретті функциялардың артықшылықтарын түсінуін бағалау.

9. Деректердің өзгермейтіндігін қамтамасыз ету үшін Python тілінде кортеждерді қалай пайдалануға болатынын түсіндіріңіз.

- Мақсаты: өзгермейтіндікке қол жеткізу үшін кортеждер сияқты өзгермейтін деректер түрлерін пайдалануды түсіну.

10. Деректердің өзгермейтіндігі параллельді және асинхронды бағдарламалауға қалай әсер етеді?

- Мақсаты: Студенттердің бағдарламалаудағы параллельді және асинхронды операцияларды жеңілдетудегі өзгермейтіндік әсерін түсінетіндігін тексеру.

Шешімге мысал

Тапсырма: «Мұздатылған сөздік» нысанын құру функциясын орындау

Python тілінде сөздіктер (`dict`) өзгермелі деректер түрі болып табылады, бұл бағдарламаның әртүрлі бөліктерінде қолданылатын сөздіктің бір данасын өзгерту кезінде жанама әсерлерге әкелуі мүмкін. Бұл тапсырманың мақсаты - кәдімгі сөздікті алатын және оның қатып қалған аналогын қайтаратын функцияны, яғни бірдей деректерді көрсететін өзгермейтін нысанды жасау.

Негізгі тапсырма нүктелері:

1. Сөздікті аргумент ретінде қабылдайтын функцияны құрастырыңыз.

2. Функция бастапқы сөздік сияқты бірдей кілт-мән жұптарын қамтитын өзгермейтін нысанды қайтаруы керек. 3. Іске асыру таза функциялық принциптерді сақтай отырып жүзеге асырылуы тиіс.

Шешімі:

Бұл мәселені шешу үшін біз Python-ның кірістірілген `frozenset` түрін пайдалана аламыз, ол кәдімгі «жиынның» өзгермейтін аналогы болып табылады. Түпнұсқа сөздіктің кілт-мән жұптарынан «frozenset» жасаймыз.

def freeze\_dict(input\_dict):

"""

Сөздікті алып, оның мазмұнының өзгермейтін көрінісін қайтарады.

"""

# Кілт-мән жұптарының тізімін жасаңыз

элементтер = input\_dict.items()

# Frozen\_dict = frozenset (элементтер) өзгермейтіндігін қамтамасыз ету үшін тізімді frozenset түріне түрлендіру

frozen\_dict қайтару

# Функцияны пайдалану мысалы

original\_dict = {"алма": "жасыл", "банан": "сары"} frozen\_dictionary = freeze\_dict(original\_dict)

басып шығару(мұздатылған\_сөздік)

Түсініктеме:

- `freeze\_dict` функциясы аргумент ретінде `input\_dict` сөздігін қабылдайды.

- `items()` әдісін қолдану арқылы біз бастапқы сөздіктен кілт-мән жұптарының тізімін аламыз.

- Содан кейін біз бұл тізімді «frozenset» түріне түрлендіреміз, бұл оны өзгермейтін етеді.

- Қайтарылған `frozenset` бастапқы сөздіктің енді өзгертуге болмайтын "мұздатылған" нұсқасы болып табылады.

Бұл шешім Python-да деректерді қалай өзгермейтін етуге болатынын және жанама әсерлер тудырмайтын немесе сыртқы күйге тәуелді таза функцияларды қалай жазу керектігін көрсетеді.