

Advanced Python Programming (4321602) - Summer 2024 Solution

Milav Dabgar

June 21, 2024

Question 1

પ્રશ્ન 1(a) [3 ગુણ]

પાયથનમાં ટપલ અને લિસ્ટ વચ્ચેનો તફાવત લખો.

જવાબ

તફાવત:

| લક્ષણ | ટપલ | લિસ્ટ |
|--------------|--------------------------------|------------------------------------|
| મ્યુટેબિલિટી | ઇમ્યુટેબલ (બદલી શકાતું નથી) | મ્યુટેબલ (બદલી શકાય છે) |
| સિન્ટેક્સ | () સાથે બનાવાય છે | [] સાથે બનાવાય છે |
| પ્રદર્શન | ઝડપી | ધીમું |
| મેથડ્સ | મર્યાદિત મેથડ્સ (count, index) | ઘણી મેથડ્સ (append, remove, વગેરે) |

- મેમરી કાર્યક્ષમ: ટપલ લિસ્ટ કરતાં ઓછી મેમરી વાપરે છે
- ઉપયોગ: સ્થિર ડેટા માટે ટપલ, ગતિશીલ ડેટા માટે લિસ્ટ

મેમરી ટ્રીક

ટપલ ટાઇટ, લિસ્ટ લૂઝ

પ્રશ્ન 1(b) [4 ગુણ]

સેટ સમજાવો અને પાયથનમાં સેટ કેવી રીતે બનાવાય છે?

જવાબ

સેટ એ પાયથનમાં અનોખા તત્વોનો અક્રમાંકિત સંગ્રહ છે.
સેટ બનાવવાની રીતો:

```
1 # ખાલી સેટ
2 my_set = set()
3
4 # તત્વો સાથે સેટ
5 fruits = {"apple", "banana", "orange"}
6
7 # લિસ્ટમાંથી સેટ
8 numbers = set([1, 2, 3, 4])
9
```

- અનોખા તત્વો: ડુપ્લિકેટની મંજૂરી નથી
- અક્રમાંકિત: તત્વોનો કોઈ ચોક્કસ ક્રમ નથી
- ઓપરેશન્સ: યુનિયન, ઇન્ટરસેક્શન, ડિફરન્સ સપોર્ટેડ

મેમરી ટ્રીક

સેટ સ્પેશિયલ - અનોખા અને અક્રમાંકિત

પ્રશ્ન 1(c) [7 ગુણ]

પાયથનમાં ડિક્શનરી એટલે શું? બે ડિક્શનરીને નવી ડિક્શનરીમાં જોડવા માટેનો પ્રોગ્રામ લખો.

જવાબ

ડિક્શનરી એ પાયથનમાં કી-વેલ્યુ પેર્સનો ક્રમાંકિત સંગ્રહ છે.
પ્રોગ્રામ:

```

1 # બે ડિક્શનરીઓ
2 dict1 = {1: 10, 2: 20}
3 dict2 = {3: 30, 4: 40}
4
5 # મેથડ 1: update() નો ઉપયોગ
6 result1 = dict1.copy()
7 result1.update(dict2)
8
9 # મેથડ 2: ** ઓપરેટરનો ઉપયોગ
10 result2 = {**dict1, **dict2}
11
12 print("પરિણામ:", result2)
13 # આઉટપુટ: {1: 10, 2: 20, 3: 30, 4: 40}
14
```

- કી-વેલ્યુ પેર્સ: દરેક તત્વમાં કી અને વેલ્યુ હોય છે
- મ્યુટેબલ: બનાવ્યા પછી બદલી શકાય છે
- ઝડપી એક્સેસ: $O(1)$ સરેરાશ સમય જટિલતા

મેમરી ટ્રીક

ડિક્શનરી ડાયનેમિક કી-વેલ્યુ સ્ટોર છે

પ્રશ્ન 1(c) OR [7 ગુણ]

પાયથનમાં લિસ્ટ એટલે શું? એક પ્રોગ્રામ લખો જે સૂચિમાંથી મહત્તમ અને ન્યૂનતમ નંબરો શોધે.

જવાબ

લિસ્ટ એ પાયથનમાં તત્વોનો ક્રમાંકિત, મ્યુટેબલ સંગ્રહ છે.
પ્રોગ્રામ:

```

1 # ઇનપુટ લિસ્ટ
2 numbers = [45, 12, 78, 23, 56, 89, 34]
3
4 # મહત્તમ અને ન્યૂનતમ શોધો
```

```

5 maximum = max(numbers)
6 minimum = min(numbers)
7
8 print(f"મહત્તમ: {maximum}")
9 print(f"ન્યૂનતમ: {minimum}")
10
11 # મેન્યુઅલ મેથડ
12 max_val = numbers[0]
13 min_val = numbers[0]
14 for num in numbers:
15     if num > max_val:
16         max_val = num
17     if num < min_val:
18         min_val = num
19

```

- ક્રમાંકિત: તત્વો ઇન્સર્શન ઓર્ડર જાળવે છે
- ઇન્ડેક્સિંગ: ઇન્ડેક્સ [0, 1, 2...] વાપરીને એક્સેસ
- બિલ્ટ-ઇન ફંક્શન્સ: min(), max(), len() ઉપલબ્ધ

મેમરી ટ્રીક

લિસ્ટ લિનિયર અને ઇન્ડેક્સડ છે

Question 2

પ્રશ્ન 2(a) [3 ગુણ]

નેસ્ટેડ ટપલને ઉદાહરણ સાથે સમજાવો.

જવાબ

નેસ્ટેડ ટપલ એ ટપલ છે જેમાં અન્ય ટપલ તત્વો તરીકે હોય છે.

ઉદાહરણ:

```

1 # નેસ્ટેડ ટપલ
2 student_data = (
3     ("John", 85, "A"),
4     ("Alice", 92, "A+"),
5     ("Bob", 78, "B")
6 )
7
8 # તત્વોને એક્સેસ કરવું
9 print(student_data[0][1]) # આઉટપુટ: 85
10 print(student_data[1][0]) # આઉટપુટ: Alice
11

```

- બહુ-પરિમાણીય: ટપલની અંદર ટપલ
- ઇન્ડેક્સિંગ: બહુવિધ ઇન્ડેક્સ [i][j] વાપરો
- ઇમ્યુટેબલ: નેસ્ટેડ તત્વો બદલી શકાતા નથી

મેમરી ટ્રીક

નેસ્ટેડ મતલબ ટપલની અંદર ટપલ

પ્રશ્ન 2(b) [4 ગુણ]

રેન્ડમ મોડ્યુલ શું છે? ઉદાહરણ સાથે સમજાવો.

જવાબ

રેન્ડમ મોડ્યુલ રેન્ડમ નંબરો જનરેટ કરે છે અને રેન્ડમ ઓપરેશન્સ કરે છે.
ઉદાહરણ:

```
1 import random
2
3 # રેન્ડમ ઇન્ટજિર
4 num = random.randint(1, 10)
5 print(f"રેન્ડમ નંબર: {num}")
6
7 # લસ્ટમાંથી રેન્ડમ પસંદગી
8 colors = ["red", "blue", "green"]
9 choice = random.choice(colors)
10 print(f"રેન્ડમ રંગ: {choice}")
11
12 # રેન્ડમ ડેસિમલ
13 decimal = random.random()
14 print(f"રેન્ડમ દશાંશ: {decimal}")
15
```

- ઇમ્પોર્ટ જરૂરી: import random
- વિવિધ ફંક્શન્સ: randint(), choice(), random()
- ઉપયોગી: ગેમ્સ, સિમ્યુલેશન, ટેસ્ટિંગ માટે

મેમરી ટ્રીક

રેન્ડમ વસ્તુઓને અણધારી બનાવે છે

પ્રશ્ન 2(c) [7 ગુણ]

પેકેજને ઇમ્પોર્ટ કરવાની વિવિધ રીતો સમજાવો. તેનું એક ઉદાહરણ આપો.

જવાબ

ઇમ્પોર્ટ મેથડ્સ:

| મેથડ | સિન્ટેક્સ | ઉપયોગ |
|-----------------|------------------------------|--------------------|
| નોર્મલ ઇમ્પોર્ટ | import package | package.function() |
| ફ્રોમ ઇમ્પોર્ટ | from package import function | function() |
| બધું ઇમ્પોર્ટ | from package import * | function() |
| એલિયાસ ઇમ્પોર્ટ | import package as alias | alias.function() |

ઉદાહરણ:

```
1 # નોર્મલ ઇમ્પોર્ટ
2 import math
3 result1 = math.sqrt(16)
4
5 # ફ્રોમ ઇમ્પોર્ટ
6 from math import sqrt
7 result2 = sqrt(16)
```

```

8
9 # એલિયાસ સાથે ઇમ્પોર્ટ
10 import math as m
11 result3 = m.sqrt(16)
12
13 # બધું ઇમ્પોર્ટ ભલામણ( નથી)
14 from math import *
15 result4 = sqrt(16)
16

```

- નેમસ્પેસ: નોર્મલ ઇમ્પોર્ટ અલગ નેમસ્પેસ રાખે છે
- ડાયરેક્ટ એક્સેસ: ફ્રોમ ઇમ્પોર્ટ ડાયરેક્ટ ફંક્શન કોલ કરવાની મંજૂરી આપે છે
- એલિયાસ: સુવિધા માટે ટૂંકા નામો

મેમરી ટ્રીક

ઇમ્પોર્ટ મેથડ્સ: નોર્મલ, ફ્રોમ, બધું, એલિયાસ

પ્રશ્ન 2(a) OR [3 ગુણ]

પાયથનમાં ડિક્શનરીના ગુણધર્મો લખો.

જવાબ

ડિક્શનરીના ગુણધર્મો:

| ગુણધર્મ | વર્ણન |
|--------------|---------------------------------------|
| ક્રમાંકિત | ઇન્સર્શન ઓર્ડર જાળવે છે (Python 3.7+) |
| મ્યુટેબલ | બનાવ્યા પછી બદલી શકાય છે |
| કી-અનોખી | ડુપ્લિકેટ કીઓની મંજૂરી નથી |
| હેટેરોજીનિયસ | કીઓ અને વેલ્યુઝ અલગ પ્રકારના હોઈ શકે |

- ઝડપી એક્સેસ: $O(1)$ સરેરાશ લુકઅપ ટાઇમ
- ડાયનેમિક સાઇઝ: વધી અથવા ઘટી શકે છે
- કી પ્રતિબંધો: કીઓ ઇમ્યુટેબલ હોવી જોઈએ

મેમરી ટ્રીક

ડિક્શનરી ક્રમાંકિત, મ્યુટેબલ, અનોખી, હેટેરોજીનિયસ છે

પ્રશ્ન 2(b) OR [4 ગુણ]

પાયથનમાં dir() ફંક્શન શું છે. ઉદાહરણ સાથે સમજાવો.

જવાબ

dir() ફંક્શન ઓબ્જેક્ટના બધા એટ્રિબ્યુટ્સ અને મેથડ્સ રિટર્ન કરે છે.
ઉદાહરણ:

```

1 # સ્ટ્રિંગના બધા એટ્રિબ્યુટ્સ
2 text = "hello"

```

```

3 attributes = dir(text)
4 print(attributes[:5]) # પ્રથમ 5 એટ્રિબ્યુટ્સ
5
6 # ઉપલબ્ધ મેથડ્સ ચેક કરો
7 print("upper" in dir(text)) # True
8
9 # મોડ્યુલ્સ માટે
10 import math
11 math_methods = dir(math)
12 print("sqrt" in math_methods) # True
13
14 # કસ્ટમ ઓબ્જેક્ટ્સ માટે
15 class MyClass:
16     def my_method(self):
17         pass
18
19 obj = MyClass()
20 print(dir(obj))
21

```

- **ઇન્ટ્રોસ્પેક્શન:** ઓબ્જેક્ટ પ્રોપર્ટીઝ તપાસે છે
- **ડિબગિંગ:** ઉપલબ્ધ મેથડ્સ શોધવામાં મદદ કરે છે
- **બધા ઓબ્જેક્ટ્સ:** કોઈપણ Python ઓબ્જેક્ટ સાથે કામ કરે છે

મેમરી ટ્રીક

dir() ઓબ્જેક્ટ એટ્રિબ્યુટ્સની ડિરેક્ટરી બતાવે છે

પ્રશ્ન 2(c) OR [7 ગુણ]

બે સંખ્યાઓનો સરવાળો શોધવા માટે મોડ્યુલને વ્યાખ્યાયિત કરવા માટે પ્રોગ્રામ લખો. બીજા પ્રોગ્રામમાં મોડ્યુલ ઇમ્પોર્ટ કરો.

જવાબ

મોડ્યુલ ફાઇલ (calculator.py):

```

1 # calculator.py
2 def add_numbers(a, b):
3     """બે સંખ્યાઓ ઉમેરવા માટેનું ફંક્શન"""
4     return a + b
5
6 def multiply_numbers(a, b):
7     """બે સંખ્યાઓ ગુણવા માટેનું ફંક્શન"""
8     return a * b
9
10 def get_sum(num1, num2):
11     """ત્રીસ વર્ગ સમ ફંક્શન"""
12     result = num1 + num2
13     return result
14

```

મુખ્ય પ્રોગ્રામ (main.py):

```

1 # main.py
2 import calculator
3
4 # મોડ્યુલનો ઉપયોગ
5 result1 = calculator.add_numbers(10, 20)

```

```

6 print(f"સરવાળો: {result1}")
7
8 # ફરોમ ઇમ્પોર્ટ
9 from calculator import get_sum
10 result2 = get_sum(15, 25)
11 print(f"ફરોમ ઇમ્પોર્ટ વાપરીને સરવાળો: {result2}")
12

```

- મોડ્યુલ બનાવટ: ફંક્શન-સને .py ફાઇલમાં સેવ કરો
- ઇમ્પોર્ટ: ઇમ્પોર્ટ સ્ટેટમેન્ટ વાપરીને એક્સેસ કરો
- કોડ પુનઃઉપયોગ: એક જ મોડ્યુલને અનેક પ્રોગ્રામમાં વાપરો

મેમરી ટ્રીક

મોડ્યુલ કોડને પુનઃઉપયોગી અને વ્યવસ્થિત બનાવે છે

Question 3

પ્રશ્ન 3(a) [3 ગુણ]

રનટાઇમ એરર અને લોજિકલ એરર શું છે. ઉદાહરણ સાથે સમજાવો.

જવાબ

તફાવત:

| એરર પ્રકાર | વ્યાખ્યા | ઉદાહરણ |
|------------|--|-------------------------------|
| રનટાઇમ એરર | પ્રોગ્રામના અમલ દરમિયાન થાય છે | ઝીરો વડે ભાગાકાર, ફાઇલ ન મળવી |
| લોજિકલ એરર | પ્રોગ્રામ ચાલે છે પણ ખોટું આઉટપુટ આપે છે | ખોટું સૂત્ર, ખોટી શરત |

ઉદાહરણો:

```

1 # રનટાઇમ એરર
2 x = 10
3 y = 0
4 result = x / y # ZeroDivisionError
5
6 # લોજિકલ એરર
7 def calculate_area(radius):
8     return 3.14 * radius # radius * radius હોવું જોઈએ
9

```

મેમરી ટ્રીક

રનટાઇમ ક્રેશ કરે, લોજિકલ કન્ફ્યુઝ કરે

પ્રશ્ન 3(b) [4 ગુણ]

Except પર પોઇન્ટ્સ લખો અને તેને સમજાવો.

જવાબ

Except ક્લોઝ try-except બ્લોકમાં ચોક્કસ એક્સેપ્શન હેન્ડલ કરે છે.
મુખ્ય પોઇન્ટ્સ:

| લક્ષણ | વર્ણન |
|-----------|---|
| સિન્ટેક્સ | except ExceptionType: |
| બહુવિધ | એકથી વધુ except બ્લોક હોઈ શકે છે |
| સામાન્ય | except: બધા એક્સેપ્શન પકડે છે |
| વેરિએબલ | except Exception as e: એરર સ્ટોર કરે છે |

```

1 try:
2     number = int(input("નંબર દાખલ કરો: "))
3     result = 10 / number
4 except ValueError:
5     print("અમાન્ય ઇનપુટ")
6 except ZeroDivisionError:
7     print("ઝીરો વડે ભાગી શકાતું નથી")
8 except Exception as e:
9     print(f"એરર: {e}")
10

```

મેમરી ટ્રીક

Except એરર પકડે છે અને હેન્ડલ કરે છે

પ્રશ્ન 3(c) [7 ગુણ]

Divide by zero Exception પકડવા માટે પ્રોગ્રામ લખો. તેમજ finally બ્લોકનો ઉપયોગ કરો.

જવાબ

પ્રોગ્રામ:

```

1 def safe_division():
2     try:
3         # યુઝર પાસેથી ઇનપુટ મેળવો
4         numerator = float(input("અંશ દાખલ કરો: "))
5         denominator = float(input("છેદ દાખલ કરો: "))
6
7         # ભાગાકાર કરો
8         result = numerator / denominator
9         print(f"પરિણામ: {numerator} / {denominator} = {result}")
10
11     except ZeroDivisionError:
12         print("એરર: ઝીરો વડે ભાગી શકાતું નથી!")
13         print("કૃપા કરીને નોનઝીરો- છેદ દાખલ કરો")
14
15     except ValueError:
16         print("એરર: કૃપા કરીને માન્ય નંબરો જ દાખલ કરો")
17
18     except Exception as e:
19         print(f"અનપેક્ષિત એરર: {e}")
20
21     finally:

```



```

22 print("ભાગાકાર પ્રક્રિયા પૂરણ થઈ")
23 print("કેલ્ક્યુલેટર વાપરવા બદલ આભાર")
24
25 # ફંક્શન કોલ કરો
26 safe_division()
27

```

- **Try બ્લોક:** જોખમી કોડ ધરાવે છે
- **Except:** ખાસ કરીને ZeroDivisionError હેન્ડલ કરે છે
- **Finally:** એક્સેપ્શન ગમે તે હોય, હંમેશા એક્ઝિક્યુટ થાય છે

મેમરી ટ્રીક

Try જોખમી કોડ, Except એરર હેન્ડલ કરે, Finally હંમેશા ચાલે

પ્રશ્ન 3(a) OR [3 ગુણ]

બિલ્ટ-ઇન એક્સેપ્શન શું છે અને તેના પ્રકારો આપો.

જવાબ

બિલ્ટ-ઇન એક્સેપ્શન પ્રકારો:

| પ્રકાર | વર્ણન | ઉદાહરણ |
|-------------------|---------------------------|-----------------|
| ValueError | ઓપરેશન માટે અમાન્ય વેલ્યુ | int("abc") |
| TypeError | ખોટો ડેટા ટાઇપ | "5" + 5 |
| IndexError | ઇન્ડેક્સ રેન્જ બહાર | list[10] |
| KeyError | કી મળી નથી | dict["missing"] |
| FileNotFoundError | ફાઇલ અસ્તિત્વમાં નથી | open("no.txt") |

મેમરી ટ્રીક

Value, Type, Index, Key, File - સામાન્ય એરર પ્રકારો

પ્રશ્ન 3(b) OR [4 ગુણ]

Syntax error સમજાવો અને આપણે તેને કેવી રીતે ઓળખી શકીએ? ઉદાહરણ આપો.

જવાબ

Syntax Error ત્યારે થાય છે જ્યારે ખોટા સિન્ટેક્સને કારણે Python કોડ પાર્સ કરી શકતું નથી.

ઓળખ:

| પદ્ધતિ | વર્ણન |
|--------------------|--|
| Python interpreter | લાઇન નંબર સાથે એરર મેસેજ બતાવે છે |
| IDE highlighting | કોડ એડિટર્સ સિન્ટેક્સ એરર હાઇલાઇટ કરે છે |
| એરર મેસેજ | એરરની ચોક્કસ જગ્યા દર્શાવે છે |

ઉદાહરણો:

```

1 # કોલોન ખૂટે છે
2 if x > 5
3     print("Greater") # SyntaxError
4
5 # કૌંસ બંધ નથી
6 print("Hello" # SyntaxError
7
8 # ખોટું ઇન્ડેન્ટેશન
9 def my_function():
10     print("Hello") # IndentationError
11

```

મેમરી ટ્રીક

Syntax એરર કોડને શરૂ થતા અટકાવે છે

પ્રશ્ન 3(c) OR [7 ગુણ]

પાયથનમાં Exception handling શું છે? યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Exception Handling એ પ્રોગ્રામ ક્રેશ થયા વગર રનટાઇમ એરરને ગ્રેસફુલી હેન્ડલ કરવાની પદ્ધતિ છે.

પ્રોગ્રામ:

```

1 def file_processor():
2     filename = None
3     try:
4         filename = input("ફાઇલનામ દાખલ કરો: ")
5         with open(filename, 'r') as file:
6             content = file.read()
7             numbers = [int(x) for x in content.split()]
8             average = sum(numbers) / len(numbers)
9             print(f"સરેરાશ: {average}")
10
11     except FileNotFoundError:
12         print(f"એરર: ફાઇલ '{filename}' મળી નથી")
13
14     except ValueError:
15         print("એરર: ફાઇલમાં નોનન્યૂમેરિક- ડેટા છે")
16
17     except ZeroDivisionError:
18         print("એરર: ફાઇલમાં કોઈ નંબરો મળ્યા નથી")
19
20     except Exception as e:
21         print(f"અનપેક્ષિત એરર: {e}")
22
23     else:
24         print("ફાઇલ સફળતાપૂર્વક પ્રોસેસ થઈ")
25
26     finally:
27         print("ફાઇલ પ્રોસેસિંગ ઓપરેશન પૂરું થયું")
28
29 # ફંક્શન ચલાવો
30 file_processor()
31

```

મેમરી ટ્રીક

Try-Except-Else-Finally: સંપૂર્ણ એરર હેન્ડલિંગ

Question 4

પ્રશ્ન 4(a) [3 ગુણ]

ફાઇલમાં આપણે કયા પ્રકારના વિવિધ ઓપરેશન્સ કરી શકીએ?

જવાબ

ફાઇલ ઓપરેશન્સ:

| ઓપરેશન | વર્ણન | મેથડ |
|--------|---------------------|--------------------|
| Read | ફાઇલ કન્ટેન્ટ વાંચો | read(), readline() |
| Write | ફાઇલમાં ડેટા લખો | write() |
| Append | અંતે ડેટા ઉમેરો | mode 'a' |
| Create | નવી ફાઇલ બનાવો | mode 'w', 'x' |
| Delete | ફાઇલ દૂર કરો | os.remove() |
| Seek | ફાઇલ પોઇન્ટર ખસેડો | seek() |

મેમરી ટ્રીક

Read, Write, Append, Create, Delete, Seek

પ્રશ્ન 4(b) [4 ગુણ]

ફાઇલ મોડ્સની યાદી આપો. કોઈપણ ચાર મોડ્સનું વર્ણન લખો.

જવાબ

ફાઇલ મોડ્સ:

| મોડ | વર્ણન | હેતુ |
|-----|------------|--|
| 'r' | Read મોડ | અસ્તિત્વમાં રહેલી ફાઇલ વાંચો (ડિફોલ્ટ) |
| 'w' | Write મોડ | ફાઇલ બનાવો/ઓવરરાઇટ કરો |
| 'a' | Append મોડ | ફાઇલના અંતે ઉમેરો |
| 'x' | Exclusive | નવી બનાવો, જો અસ્તિત્વમાં હોય તો ફેલ |
| 'b' | Binary મોડ | બાઈનરી ફાઇલો |
| '+' | Read+Write | અપડેટ મોડ |

મેમરી ટ્રીક

Read, Write, Append, eXclusive - મુખ્ય ફાઇલ મોડ્સ

પ્રશ્ન 4(c) [7 ગુણ]

ફાઇલમાંના તમામ શબ્દોને સોર્ટ કરવા અને તેને લિસ્ટમાં મૂકવા માટેનો પ્રોગ્રામ લખો.

જવાબ

પ્રોગ્રામ:

```

1 def sort_words_from_file():
2     try:
3         # ઇનપુટ ફાઇલનામ
4         filename = input("ફાઇલનામ દાખલ કરો: ")
5
6         # ફાઇલ કન્ટેન્ટ વાંચો
7         with open(filename, 'r') as file:
8             content = file.read()
9
10        # વર્ણિજિત કરો અને સાફ કરો
11        words = content.lower().split()
12
13        # Punctuation દૂર કરો
14        import string
15        clean_words = []
16        for word in words:
17            clean_word = word.translate(str.maketrans("", "", string.punctuation))
18            if clean_word: # માત્ર ખાલી ન હોય તેવા શબ્દો
19                clean_words.append(clean_word)
20
21        # શબ્દો સોર્ટ કરો
22        sorted_words = sorted(clean_words)
23
24        # પરિણામો દર્શાવો
25        print("સોર્ટ થયેલા શબ્દો:")
26        print(sorted_words)
27
28        # નવી ફાઇલમાં સેવ કરો
29        with open('sorted_words.txt', 'w') as output_file:
30            for word in sorted_words:
31                output_file.write(word + '\n')
32
33        print(f"કુલ શબ્દો: {len(sorted_words)}")
34
35    except FileNotFoundError:
36        print("એરર: ફાઇલ મળી નથી")
37    except Exception as e:
38        print(f"એરર: {e}")
39
40    sort_words_from_file()
41
```

મેમરી ટ્રીક

Read, Split, Clean, Sort, Save

પ્રશ્ન 4(a) OR [3 ગુણ]

ફાઇલ ડેન્ડલિંગ શું છે? ફાઇલ ડેન્ડલિંગ ઓપરેશનની યાદી આપો અને સમજાવો.

જવાબ

ફાઇલ હેન્ડલિંગ એ ડેટાને કાયમી ધોરણે સ્ટોર કરવા અને પુનઃપ્રાપ્ત કરવા માટે ફાઇલો સાથે કામ કરવાની પ્રક્રિયા છે.
ફાઇલ હેન્ડલિંગ ઓપરેશન્સ:

| ઓપરેશન | ફંક્શન | વર્ણન |
|--------|---------|------------------------------|
| Open | open() | ચોક્કસ મોડમાં ફાઇલ ખોલે છે |
| Read | read() | ફાઇલમાંથી ડેટા વાંચે છે |
| Write | write() | ફાઇલમાં ડેટા લખે છે |
| Close | close() | ફાઇલ બંધ કરે છે |
| Seek | seek() | ફાઇલ પોઇન્ટર ખસેડે છે |
| Tell | tell() | વર્તમાન પોઝિશન રિટર્ન કરે છે |

મેમરી ટ્રીક

Open, Read, Write, Close - બેઝિક ફાઇલ સાયકલ

પ્રશ્ન 4(b) OR [4 ગુણ]

load() મેથડ ઉદાહરણ સાથે સમજાવો.

જવાબ

load() મેથડ ફાઇલમાંથી ડેટાને ડીસીરિયલાઇઝ કરવા માટે વપરાય છે (સામાન્ય રીતે pickle મોડ્યુલ સાથે).
ઉદાહરણ:

```

1 import pickle
2
3 # પહેલાં, ડેટા સેવ કરીએ
4 data_to_save = {'name': 'John', 'scores': [85, 92, 78]}
5 with open('data.pkl', 'wb') as file:
6     pickle.dump(data_to_save, file)
7
8 # ફાઇલમાંથી ડેટા લોડ કરીએ
9 with open('data.pkl', 'rb') as file:
10     loaded_data = pickle.load(file)
11
12 print("લોડ થયેલ ડેટા:", loaded_data)
13
```

- ડીસીરિયલાઇઝેશન: ફાઇલ ડેટાને પાછું Python objects માં કન્વર્ટ કરે છે
- Binary મોડ: pickle ફાઇલ્સ માટે 'rb' મોડ વાપરો

મેમરી ટ્રીક

load() ફાઇલ ડેટાને પાછું Python objects માં લાવે છે

પ્રશ્ન 4(c) OR [7 ગુણ]

એક પ્રોગ્રામ લખો જે ટેક્સ્ટ ફાઇલને ઇનપુટ કરે. પ્રોગ્રામે ફાઇલમાંના તમામ યુનિક શબ્દોને મૂળાક્ષરોના ક્રમમાં છાપવા જોઈએ.

જવાબ

પ્રોગ્રામ:

```

1 def find_unique_words():
2     try:
3         # ફાઇલનામ મેળવો
4         filename = input("ટેક્સ્ટ ફાઇલનામ દાખલ કરો: ")
5
6         # ફાઇલ કન્ટેન્ટ વાંચો
7         with open(filename, 'r', encoding='utf-8') as file:
8             content = file.read().lower()
9
10        # સાફ કરો અને શબ્દો એક્સ્ટ્રેક્ટ કરો
11        import re
12        words = re.findall(r'\b[a-zA-Z]+\b', content)
13
14        # સેટ યુનિક() બનાવો અને સોર્ટ કરો
15        unique_words = sorted(list(set(words)))
16
17        # પરિણામો દર્શાવો
18        print("\nમૂળાકૃષ્ણરોનન ક્રમમાં યુનિક શબ્દો:")
19        for i, word in enumerate(unique_words, 1):
20            print(f"{i:3d}. {word}")
21
22        print(f"\nકુલન યુનિક શબ્દો: {len(unique_words)}")
23
24        # પરિણામો સેવ કરો
25        with open('unique_words.txt', 'w') as f:
26            for word in unique_words:
27                f.write(word + '\n')
28
29        except FileNotFoundError:
30            print(f"એરર: ફાઇલ '{filename}' મળી નથી")
31        except Exception as e:
32            print(f"એરર: {e}")
33
34    find_unique_words()
35

```

મેમરી ટ્રીક

વાંચો, એક્સ્ટ્રેક્ટ કરો, યુનિક, સોર્ટ, દર્શાવો

Question 5

પ્રશ્ન 5(a) [3 ગુણ]

નીચેના ટર્ટલ ફંક્શનને યોગ્ય ઉદાહરણ સાથે સમજાવો. (a) turn() (b) move().

જવાબ

નોંધ: સ્ટાન્ડર્ડ ટર્ટલ left/right (turn) અને forward/backward (move) વાપરે છે.
ફંક્શન:

| ફંક્શન | હેતુ | ઉદાહરણ |
|--------|----------------|---------------------|
| Turn | દિશા બદલે છે | turtle.left(90) |
| Move | પોઝિશન બદલે છે | turtle.forward(100) |

```

1 import turtle
2 t = turtle.Turtle()
3 t.forward(100) # Move
4 t.left(90)    # Turn
5

```

મેમરી ટ્રીક

Turn દિશા બદલે, Move પોઝિશન બદલે

પ્રશ્ન 5(b) [4 ગુણ]

ટર્ટલની દિશા બદલવાની વિવિધ ઇનબિલ્ટ પદ્ધતિઓ સમજાવો.

જવાબ

દિશા મેથડ્સ:

| મેથડ | વર્ણન | ઉદાહરણ |
|-----------------|--------------------------|-----------------|
| left(deg) | ડાબે ફેરવો (વામાવર્ત) | t.left(90) |
| right(deg) | જમણે ફેરવો (દક્ષિણાવર્ત) | t.right(45) |
| setheading(deg) | ચોક્કસ કોણ સેટ કરો | t.setheading(0) |
| towards(x,y) | પોઇન્ટ તરફનો કોણ | t.towards(0,0) |

મેમરી ટ્રીક

Left-Right સંબંધિત, Heading ચોક્કસ, Towards ગણતરી કરે

પ્રશ્ન 5(c) [7 ગુણ]

ટર્ટલનો ઉપયોગ કરીને ચોરસ, લંબચોરસ અને વર્તુળ દોરવા માટેનો પ્રોગ્રામ લખો.

જવાબ

પ્રોગ્રામ:

```

1 import turtle
2
3 def draw_shapes():
4     t = turtle.Turtle()
5     t.speed(3)
6
7     # ચોરસ દોરો
8     t.penup(); t.goto(-200, 50); t.pendown()
9     t.write("ચોરસ")

```

```

10 for _ in range(4):
11     t.forward(80)
12     t.right(90)
13
14 # લંબચોરસ દોરો
15 t.penup(); t.goto(0, 50); t.pendown()
16 t.write("લંબચોરસ")
17 for _ in range(2):
18     t.forward(120) # લંબાઈ
19     t.right(90)
20     t.forward(60) # પહોળાઈ
21     t.right(90)
22
23 # વર્તુળ દોરો
24 t.penup(); t.goto(200, 50); t.pendown()
25 t.write("વર્તુળ")
26 t.circle(40)
27
28 turtle.done()
29
30 draw_shapes()
31

```

મેમરી ટ્રીક

ચોરસ: 4 સમાન બાજુ, લંબચોરસ: 2 જોડી, વર્તુળ: radius મેથડ

પ્રશ્ન 5(a) OR [3 ગુણ]

ટર્ટલમાં પેન કમાન્ડના વિવિધ પ્રકારો કયા છે? તે બધાને સમજાવો.

જવાબ

પેન કમાન્ડ્સ:

| કમાન્ડ | હેતુ |
|--------------|------------------------------|
| penup() | પેન ઉઠાવો (દોરવાનું બંધ) |
| pendown() | પેન નીચે મૂકો (દોરવાનું શરૂ) |
| pensize(w) | લાઇન જાડાઈ સેટ કરો |
| pencolor(c) | લાઇન કલર સેટ કરો |
| fillcolor(c) | ભરવાનો કલર સેટ કરો |
| begin_fill() | ભરવાનું શરૂ કરો |
| end_fill() | ભરવાનું બંધ કરો |

મેમરી ટ્રીક

Up-Down દોરવાનું કન્ટ્રોલ કરે, Size-Color દેખાવ કન્ટ્રોલ કરે

પ્રશ્ન 5(b) OR [4 ગુણ]

ટર્ટલનો ઉપયોગ કરીને વર્તુળ અને સ્ટારના આકાર દોરો અને તેમને લાલ રંગથી ભરો.

જવાબ

પ્રોગ્રામ:

```

1 import turtle
2
3 t = turtle.Turtle()
4 t.color("red", "red") # પેન અને ફિલ કલર
5
6 # ભરેલું વસ્તુ
7 t.begin_fill()
8 t.circle(50)
9 t.end_fill()
10
11 t.penup(); t.forward(150); t.pendown()
12
13 # ભરેલો સ્પષ્ટાર
14 t.begin_fill()
15 for _ in range(5):
16     t.forward(100)
17     t.right(144)
18 t.end_fill()
19
20 turtle.done()
21

```

મેમરી ટ્રીક

Begin fill, આકાર દોરો, End fill

પ્રશ્ન 5(c) OR [7 ગુણ]

ટર્ટલનો ઉપયોગ કરીને ભારતનો જંડો દોરવા માટેનો પ્રોગ્રામ લખો.

જવાબ

ભારતનો જંડો પ્રોગ્રામ:

```

1 import turtle
2
3 def draw_rect(color, x, y, width, height):
4     t.penup(); t.goto(x, y); t.pendown()
5     t.color(color)
6     t.begin_fill()
7     for _ in range(2):
8         t.forward(width); t.right(90)
9         t.forward(height); t.right(90)
10    t.end_fill()
11
12 t = turtle.Turtle()
13 t.speed(5)
14 width = 300
15 height = 60
16
17 # પટ્ટીઓ દોરો
18 draw_rect("orange", -150, 150, width, height)
19 draw_rect("white", -150, 90, width, height)
20 draw_rect("green", -150, 30, width, height)

```

```
21
22 # ચક્કર દોરો
23 t.penup()
24 t.goto(0, 60) # સફેદ પટ્ટીનું કેન્દ્ર
25 t.pendown()
26 t.color("navy")
27 t.circle(30) # બાહ્ય વર્તુળ
28
29 # તીલીઓ
30 for i in range(24):
31     t.penup(); t.goto(0, 90); t.pendown()
32     t.setheading(i * 15)
33     t.forward(30)
34
35 t.hideturtle()
36 turtle.done()
37
```

મેમરી ટ્રીક

કેસરી-સફેદ-લીલી પટ્ટીઓ 24-તીલીવાળા ચક્ર સાથે