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:

```
# ખાલી સેટ
my_set = set()

# તત્વો સાથે સેટ
fruits = {"apple", "banana", "orange"}

# લિસ્ટમાંથી સેટ
numbers = set([1, 2, 3, 4])

```

- **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:

```
# બે ડક્ષ્નરીઓ
dict1 = {1: 10, 2: 20}
dict2 = {3: 30, 4: 40}

# મેથડ 1: update() નો ઉપયોગ
result1 = dict1.copy()
result1.update(dict2)

# મેથડ 2: ** ઓપરેટરનો ઉપયોગ
result2 = {**dict1, **dict2}

print("પરણિામ:", result2)
# આઉટપુટ: {1: 10, 2: 20, 3: 30, 4: 40}

```

- **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:

```
# ઇનપુટ લસ્ટિ
numbers = [45, 12, 78, 23, 56, 89, 34]

# મહત્તમ અને ન્યૂનતમ શોધો
```

```
maximum = max(numbers)
minimum = min(numbers)

print(f"મહત્તમ: {maximum}")
print(f"ન્યૂનતમ: {minimum}")

# મેન્યુઅલ મેથડ
max_val = numbers[0]
min_val = numbers[0]
for num in numbers:
    if num > max_val:
        max_val = num
    if num < min_val:
        min_val = num

```

- **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 ઇન્ડેક્સ્ડ 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:

```
# નેસ્ટેડ ટપલ
student_data = (
    ("John", 85, "A"),
    ("Alice", 92, "A+"),
    ("Bob", 78, "B")
)

# તત્વોને એક્સેસ કરવું
print(student_data[0][1])  # આઉટપુટ: 85
print(student_data[1][0])  # આઉટપુટ: Alice

```

- **બહુ-પરિમાણીય**: ટપલની અંદર ટપલ
- **ઇન્ડેક્સિંગ**: બહુવિધ ઇન્ડિસેસ [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:

```
import random

# રેન્ડમ ઇન્ટજિર
num = random.randint(1, 10)
print(f"રેન્ડમ નંબર: {num}")

# લસ્ટિમાંથી રેન્ડમ પસંદગી
colors = ["red", "blue", "green"]
choice = random.choice(colors)
print(f"રેન્ડમ રંગ: {choice}")

# રેન્ડમ ફ્લોટ
decimal = random.random()
print(f"રેન્ડમ દશાંશ: {decimal}")

```

- **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:

```
# નોર્મલ ઇમ્પોર્ટ
import math
result1 = math.sqrt(16)

# ફ્રોમ ઇમ્પોર્ટ
from math import sqrt
result2 = sqrt(16)
```

```
# એલિયાસ સાથે ઇમ્પોર્ટ
import math as m
result3 = m.sqrt(16)

# બધું ઇમ્પોર્ટ ભલામણ( નથી)
from math import *
result4 = sqrt(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:

```
# સ્ટ્રગિના બધા એટ્રબ્યુટ્સ
text = "hello"
```

```
attributes = dir(text)
print(attributes[:5])  # પ્રથમ 5 એટ્રિબ્યુટ્સ

# ઉપલબ્ધ મેથડ્સ ચેક કરો
print("upper" in dir(text))  # True

# મોડ્યુલ્સ માટે
import math
math_methods = dir(math)
print("sqrt" in math_methods)  # True

# કસ્ટમ ઓબ્જેક્ટ્સ માટે
class MyClass:
    def my_method(self):
        pass

obj = MyClass()
print(dir(obj))

```

- **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):

```
# calculator.py
def add_numbers(a, b):
    """બે સંખ્યાઓ ઉમેરવા માટેનું ફંક્શન"""
    return a + b

def multiply_numbers(a, b):
    """બે સંખ્યાઓ ગુણવા માટેનું ફંક્શન"""
    return a * b

def get_sum(num1, num2):
    """વૈકલ્પિક સમ ફંક્શન"""
    result = num1 + num2
    return result

```

મુખ્ય પ્રોગ્રામ (main.py):

```
# main.py
import calculator

# મોડ્યુલનો ઉપયોગ
result1 = calculator.add_numbers(10, 20)
```

```
print(f"સરવાળો: {result1}")

# ફૂરોમ ઇમ્પોર્ટ
from calculator import get_sum
result2 = get_sum(15, 25)
print(f"ફૂરોમ ઇમ્પોર્ટ વાપરીને સરવાળો: {result2}")

```

- **મોડ્યુલ બનાવટ**: ફંક્શન્સને .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:

```
# રનટાઇમ એરર
x = 10
y = 0
result = x / y  # ZeroDivisionError

# લોજકિલ એરર
def calculate_area(radius):
    return 3.14 * radius  # radius * radius હોવું જોઈએ

```

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

```
try:
    number = int(input("નંબર દાખલ કરો: "))
    result = 10 / number
except ValueError:
    print("અમાન્ય ઇનપુટ")
except ZeroDivisionError:
    print("ઝીરો વડે ભાગી શકાતું નથી")
except Exception as e:
    print(f"એરર: {e}")

```

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:

```
def safe_division():
    try:
        # યુઝર પાસેથી ઇનપુટ મેળવો
        numerator = float(input("અંશ દાખલ કરો: "))
        denominator = float(input("છેદ દાખલ કરો: "))

        # ભાગાકાર કરો
        result = numerator / denominator
        print(f"પરિણામ: {numerator} / {denominator} = {result}")

    except ZeroDivisionError:
        print("એરર: ઝીરો વડે ભાગી શકાતું નથી!")
        print("કૃપા કરીને નોનઝીરો- છેદ દાખલ કરો")

    except ValueError:
        print("એરર: કૃપા કરીને માન્ય નંબરો જ દાખલ કરો")

    except Exception as e:
        print(f"અનપેક્ષિત એરર: {e}")

    finally:
```

```
        print("ભાગાકાર પ્રક્રિયા પૂર્ણ થઈ")
        print("કેલ્ક્યુલેટર વાપરવા બદલ આભાર")

# ફંક્શન કોલ કરો
safe_division()

```

- **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:

```
# કોલોન ખૂટે છે
if x > 5
    print("Greater")  # SyntaxError

# કૌંસ બંધ નથી
print("Hello"  # SyntaxError

# ખોટું ઇન્ડેન્ટેશન
def my_function():
print("Hello")  # IndentationError

```

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:

```
def file_processor():
    filename = None
    try:
        filename = input("ફાઇલનામ દાખલ કરો: ")
        with open(filename, 'r') as file:
            content = file.read()
            numbers = [int(x) for x in content.split()]
            average = sum(numbers) / len(numbers)
            print(f"સરેરાશ: {average}")

    except FileNotFoundError:
        print(f"એરર: ફાઇલ '{filename}' મળી નથી")

    except ValueError:
        print("એરર: ફાઇલમાં નોનન્યૂમેરકિ- ડેટા છે")

    except ZeroDivisionError:
        print("એરર: ફાઇલમાં કોઈ નંબરો મળ્યા નથી")

    except Exception as e:
        print(f"અનપેક્ષતિ એરર: {e}")

    else:
        print("ફાઇલ સફળતાપૂર્વક પ્રોસેસ થઈ")

    finally:
        print("ફાઇલ પ્રોસેસગિ ઓપરેશન પૂર્ણ થયું")

# ફંક્શન ચલાવો
file_processor()

```

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 મોડનું 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 સૉર્ટ કરવા અને તેને લિસ્ટમાં મૂકવા માટેનો પ્રોગ્રામ લખો.

જવાબ

પ્રોગ્રામ:

```
def sort_words_from_file():
    try:
        # ઇનપુટ ફાઇલનામ
        filename = input("ફાઇલનામ દાખલ કરો: ")

        # ફાઇલ કન્ટેન્ટ વાંચો
        with open(filename, 'r') as file:
            content = file.read()

        # વિભાજિત કરો અને સાફ કરો
        words = content.lower().split()

        # Punctuation દૂર કરો
        import string
        clean_words = []
        for word in words:
            clean_word = word.translate(str.maketrans('', '', string.punctuation))
            if clean_word:  # માત્ર ખાલી ન હોય તેવા શબ્દો
                clean_words.append(clean_word)

        # શબ્દો સૉર્ટ કરો
        sorted_words = sorted(clean_words)

        # પરિણામો દર્શાવો
        print("સૉર્ટ થયેલા શબ્દો:")
        print(sorted_words)

        # નવી ફાઇલમાં સેવ કરો
        with open('sorted_words.txt', 'w') as output_file:
            for word in sorted_words:
                output_file.write(word + '\n')

        print(f"કુલ શબ્દો: {len(sorted_words)}")

    except FileNotFoundError:
        print("એરર: ફાઇલ મળી નથી")
    except Exception as e:
        print(f"એરર: {e}")

sort_words_from_file()

```

મેમરી ટ્રીક

Read, Split, Clean, Sort, Save

પ્રશ્ન 4(a) OR [3 ગુણ]

ફાઇલ હેન્ડલિંગ શું છે? ફાઇલ હેન્ડલિંગ 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:

```
import pickle

# પહેલાં, ડેટા સેવ કરીએ
data_to_save = {'name': 'John', 'scores': [85, 92, 78]}
with open('data.pkl', 'wb') as file:
    pickle.dump(data_to_save, file)

# ફાઇલમાંથી ડેટા લોડ કરીએ
with open('data.pkl', 'rb') as file:
    loaded_data = pickle.load(file)

print("લોડ થયેલ ડેટા:", loaded_data)

```

- **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 યુનીક શબ્દોને મૂળાક્ષરોના ક્રમમાં છાપવા જોઈએ.

જવાબ

પ્રોગ્રામ:

```
def find_unique_words():
    try:
        # ફાઇલનામ મેળવો
        filename = input("ટેક્સ્ટ ફાઇલનામ દાખલ કરો: ")

        # ફાઇલ કન્ટેન્ટ વાંચો
        with open(filename, 'r', encoding='utf-8') as file:
            content = file.read().lower()

        # સાફ કરો અને શબ્દો એક્સ્ટ્રેક્ટ કરો
        import re
        words = re.findall(r'\b[a-zA-Z]+\b', content)

        # સેટ યુનીક() બનાવો અને સૉર્ટ કરો
        unique_words = sorted(list(set(words)))

        # પરણિામો દર્શાવો
        print("\મૂળાક્ષરોનાn ક્રમમાં યુનીક શબ્દો:")
        for i, word in enumerate(unique_words, 1):
            print(f"{i:3d}. {word}")

        print(f"\કુલn યુનીક શબ્દો: {len(unique_words)}")

        # પરણિામો સેવ કરો
        with open('unique_words.txt', 'w') as f:
            for word in unique_words:
                f.write(word + '\n')

    except FileNotFoundError:
        print(f"એરર: ફાઇલ '{filename}' મળી નથી")
    except Exception as e:
        print(f"એરર: {e}")

find_unique_words()

```

મેમરી ટ્રીક

વાંચો, એકસ્ટ્રેક્ટ કરો, યુનીક, સૉર્ટ, 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.
ફંક્શન્સ:

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

```
import turtle
t = turtle.Turtle()
t.forward(100)  # Move
t.left(90)      # Turn

```

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:

```
import turtle

def draw_shapes():
    t = turtle.Turtle()
    t.speed(3)

    # ચોરસ દોરો
    t.penup(); t.goto(-200, 50); t.pendown()
    t.write("ચોરસ")
```

```
    for _ in range(4):
        t.forward(80)
        t.right(90)

    # લંબચોરસ દોરો
    t.penup(); t.goto(0, 50); t.pendown()
    t.write("લંબચોરસ")
    for _ in range(2):
        t.forward(120)  # લંબાઈ
        t.right(90)
        t.forward(60)   # પહોળાઈ
        t.right(90)

    # વર્તુળ દોરો
    t.penup(); t.goto(200, 50); t.pendown()
    t.write("વર્તુળ")
    t.circle(40)

    turtle.done()

draw_shapes()

```

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:

```
import turtle

t = turtle.Turtle()
t.color("red", "red")  # પેન અને ફિલ કલર

# ભરેલું વર્તુળ
t.begin_fill()
t.circle(50)
t.end_fill()

t.penup(); t.forward(150); t.pendown()

# ભરેલો સ્ટાર
t.begin_fill()
for _ in range(5):
    t.forward(100)
    t.right(144)
t.end_fill()

turtle.done()

```

મેમરી ટ્રીક

Begin fill, આકાર દોરો, End fill

પ્રશ્ન 5(c) OR [7 ગુણ]

ટર્ટલનો ઉપયોગ કરીને ભારતનો ઝંડો દોરવા માટેનો પ્રોગ્રામ લખો.

જવાબ

ભારતનો ઝંડો પ્રોગ્રામ:

```
import turtle

def draw_rect(color, x, y, width, height):
    t.penup(); t.goto(x, y); t.pendown()
    t.color(color)
    t.begin_fill()
    for _ in range(2):
        t.forward(width); t.right(90)
        t.forward(height); t.right(90)
    t.end_fill()

t = turtle.Turtle()
t.speed(5)
width = 300
height = 60

# પટ્ટીઓ દોરો
draw_rect("orange", -150, 150, width, height)
draw_rect("white", -150, 90, width, height)
draw_rect("green", -150, 30, width, height)
```

```
# ચક્ર દોરો
t.penup()
t.goto(0, 60)  # સફેદ પટ્ટીનું કેન્દ્ર
t.pendown()
t.color("navy")
t.circle(30)   # બાહ્ય વર્તુળ

# તીલીઓ
for i in range(24):
    t.penup(); t.goto(0, 90); t.pendown()
    t.setheading(i * 15)
    t.forward(30)

t.hideturtle()
turtle.done()
```

મેમરી ટ્રીક

કેસરી-સફેદ-લીલી પટ્ટીઓ 24-તીલીવાળા ચક્ર સાથે