OOPS & Python Programming (4351108) - Winter 2023 Solution

Milav Dabgar

December 06, 2023

પ્રશ્ન 1(અ) [3 ગુણ]

પાયથન પ્રોગ્રામિંગ લેન્ગવેજના કોઈ પણ 6 ઉપયોગો લખો.

જવાબ

કોષ્ટક 1. પાયથનના ઉપયોગો

ઉપયોગ ક્ષેત્ર	વર્ણન
વેબ ડેવેલપમેન્ટ	Django, Flask frameworks
ડેટા સાયન્સ	Analysis અને visualization
મશીન લર્નિંગ	AI model development
ડેસ્કટોપ એપ્લિકેશન	GUI using Tkinter, PyQt
ગેમ ડેવેલપમેન્ટ	Pygame library
ઑટોમેશન	Scripting અને testing

મેમરી ટ્રીક

``Web Data Machine Desktop Game Auto''

પ્રશ્ન 1(બ) [4 ગુણ]

પાયથન પ્રોગ્રામિંગ લેન્ગવેજની કોઈ પણ 8 વિશેષતાઓ લખો.

જવાબ

કોષ્ટક 2. પાયથનની વિશેષતાઓ

વિશેષતા	વર્ણન
સરળ સિન્ટેક્સ	વાંચવા અને લખવામાં સરળ
ઇન્ટરપ્રિટેડ	Compilation ની જરૂર નથી
ઑબ્જેક્ટ-ઓરિએન્ટેડ	OOP concepts સપોર્ટ કરે છે
ડાયનેમિક ટાઇપિંગ	Variables ને type declaration જરૂરી નથી
ક્રોસ-પ્લેટફોર્મ	Multiple OS પર ચાલે છે
મોટી લાઇબ્રેરીઓ	Rich standard library
ઓપન સોર્સ	ઉપયોગ અને modify કરવા માટે મફત
ઇન્ટરેક્ટિવ	REPL environment

મેમરી ટ્રીક

``Simple Interpreted Object Dynamic Cross Large Open Interactive''

પ્રશ્ન 1(ક) [7 ગુણ]

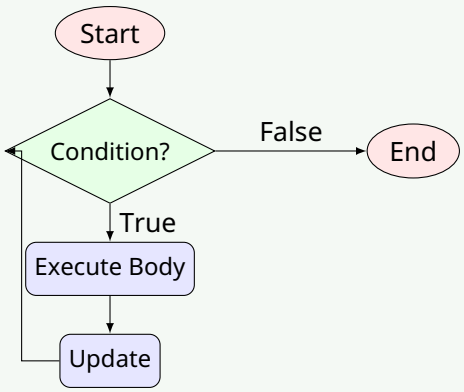
પાયથનની for અને while લૂપનું કાર્ય સમજાવો.

જવાબ

For Loop:
- **પુનરાવર્તન**: Sequences પર પુનરાવર્તન કરે છે (lists, strings, ranges)
- **સિન્ટેક્સ**: for variable in sequence:
- **આપોઆપ**: Iteration આપોઆપ handle કરે છે

While Loop:
- **શરત આધારિત**: જ્યાં સુધી શરત સાચી રહે છે
- **મેન્યુઅલ નિયંત્રણ**: Programmer iteration નો નિયંત્રણ કરે છે
- **જોખમ**: શરત કદી false ન બને તો infinite loop બની શકે છે

લૂપ ડાયાગ્રામ:



આકૃતિ 1. General Loop Flow

કોડ ઉદાહરણ:

```
# For loop
for i in range(5):
    print(i)

# While loop
i = 0
while i < 5:
    print(i)
    i += 1
```

મેમરી ટ્રીક

``For Automatic, While Manual''

પ્રશ્ન 1(ક OR) [7 ગુણ]

પાયથનના break, continue અને pass સ્ટેટમેન્ટના કાર્ય સમજાવો.

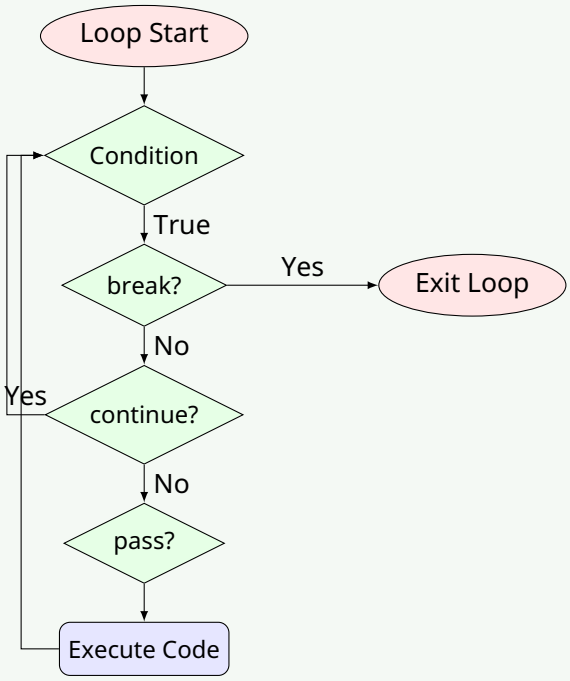
જવાબ

Break Statement:
- **બહાર નીકળવું**: સંપૂર્ણ loop ને terminate કરે છે
- **ઉપયોગ**: જ્યારે કોઈ specific condition મળે છે
- **અસર**: Control loop પછીના statement પર જાય છે

Continue Statement:
- **છોડીને આગળ**: ફક્ત current iteration skip કરે છે
- **ઉપયોગ**: Iteration માં specific values skip કરવા માટે
- **અસર**: Next iteration પર જાય છે

Pass Statement:
- **Placeholder**: કંઈ કરતું નથી, syntactic placeholder
- **ઉપયોગ**: જ્યારે syntax statement જોઈએ પણ કોઈ action નહીં
- **અસર**: કોઈ operation perform કરતું નથી

Control Flow Visualization:



આકૃતિ 2. Loop Control Statements

કોડ ઉદાહરણો:

```
# Break
for i in range(10):
    if i == 5: break
    print(i)  # 0,1,2,3,4

# Continue
for i in range(5):
    if i == 2: continue
    print(i)  # 0,1,3,4

# Pass
if True: pass  # placeholder
```

મેમરી ટ્રીક

``Break Exits, Continue Skips, Pass Waits''

પ્રશ્ન 2(અ) [3 ગુણ]

લિસ્ટના દરેક ઘટકનું મૂલ્ય 1 થી વધારવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# Method 1 - Using for loop
numbers = [1, 2, 3, 4, 5]
for i in range(len(numbers)):
    numbers[i] += 1
print(numbers)

# Method 2 - List comprehension
numbers = [1, 2, 3, 4, 5]
result = [x + 1 for x in numbers]
print(result)
```

મેમરી ટ્રીક

``Loop Index or Comprehension''

પ્રશ્ન 2(બ) [4 ગુણ]

વપરાશકર્તા પાસેથી 3 સંખ્યા લઈ તેની સરેરાશ શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# Input three numbers
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))

# Calculate average
average = (num1 + num2 + num3) / 3

# Display result
print(f"Average is: {average}")
```

મુખ્ય મુદ્દાઓ:

- **ઇનપુટ:** દશાંશ સંખ્યાઓ માટે float() ઉપયોગ કરો
- **સૂત્ર:** બધાનો સરવાળો કરીને સંખ્યા વડે ભાગો
- **આઉટપુટ:** Formatting માટે f-string ઉપયોગ કરો

મેમરી ટ્રીક

``Input Float, Sum Divide, Format Output''

પ્રશ્ન 2(ક) [7 ગુણ]

પાયથનનો list ડેટા ટાઈપ વિસ્તારથી સમજાવો.

જવાબ

લિસ્ટની લાક્ષણિકતાઓ:

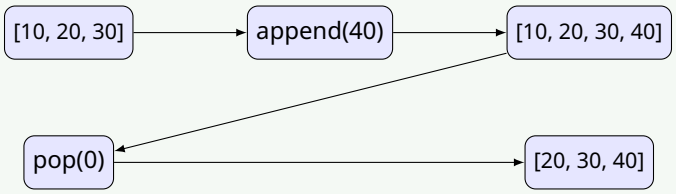
- **ક્રમબદ્ધ**: Elements sequence જાળવે છે
- **બદલાવ પાત્ર**: બનાવ્યા પછી modify કરી શકાય છે
- **વિવિધ પ્રકારની**: વિવિધ data types store કરી શકે છે
- **ઇન્ડેક્સવાળી**: Index વડે elements ને access કરી શકાય છે (0-based)

લિસ્ટ ઑપરેશન્સ ટેબલ:

કોષ્ટક 3. List Operations

ઑપરેશન	Syntax	વર્ણન
બનાવવું	list = [1,2,3]	નવી list બનાવો
એક્સેસ	list[0]	Index વડે element મેળવો
Append	list.append(4)	અંતે element ઉમેરો
Insert	list.insert(1,5)	Specific position પર ઉમેરો
Remove	list.remove(2)	પહેલું occurrence દૂર કરો
Pop	list.pop()	છેલ્લું element દૂર કરીને return કરો
Slice	list[1:3]	Sublist મેળવો

List Structure Diagram:



આકૃતિ 3. List Mutation

કોડ ઉદાહરણ:

```
# List creation and operations
fruits = ['apple', 'banana', 'orange']
fruits.append('mango')
fruits.insert(1, 'grape')
print(fruits[0])  # apple
print(len(fruits))  # 5
```

મેમરી ટ્રીક

``Ordered Mutable Heterogeneous Indexed''

પ્રશ્ન 2(અ OR) [3 ગુણ]

for લૂપની મદદથી લિસ્ટના દરેક ઘટકનો સરવાળો શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# Method 1 - Traditional for loop
numbers = [10, 20, 30, 40, 50]
total = 0
for num in numbers:
```

```
    total += num
print(f"Sum is: {total}")

# Method 2 - Using range and index
numbers = [10, 20, 30, 40, 50]
total = 0
for i in range(len(numbers)):
    total += numbers[i]
print(f"Sum is: {total}")
```

મેમરી ટ્રીક

``Initialize Zero, Loop Add, Print Total''

પ્રશ્ન 2(બ OR) [4 ગુણ]

વપરાશકર્તા પાસેથી લીધેલા મૂડલ, વ્યાજદર અને વર્ષ પરથી સાદું વ્યાજ શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# Get input from user
principal = float(input("Enter principal amount: "))
rate = float(input("Enter rate of interest: "))
time = float(input("Enter time in years: "))

# Calculate simple interest
simple_interest = (principal * rate * time) / 100

# Display results
print(f"Principal: {principal}")
print(f"Rate: {rate}%")
print(f"Time: {time} years")
print(f"Simple Interest: {simple_interest}")
print(f"Total Amount: {principal + simple_interest}")
```

સૂત્ર:

- **સાદું વ્યાજ** = (P × R × T) / 100
- **કુલ રકમ** = મૂડલ + સાદું વ્યાજ

મેમરી ટ્રીક

``Principal Rate Time, Multiply Divide Hundred''

પ્રશ્ન 2(ક OR) [7 ગુણ]

પાયથનનો tuple ડેટા ટાઈપ વિસ્તારથી સમજાવો.

જવાબ

ટ્યૂપલની લાક્ષણિકતાઓ:

- **ક્રમબદ્ધ**: Elements sequence જાળવે છે

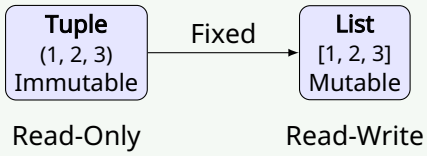
- **અપરિવર્તનીય**: બનાવ્યા પછી modify કરી શકાતું નથી
- **વિવિધ પ્રકારની**: વિવિધ data types store કરી શકે છે
- **ઇન્ડેક્સવાળી**: Index વડે access કરી શકાય છે (0-based)

ટ્યૂપલ ઑપરેશન્સ ટેબલ:

કોષ્ટક 4. Tuple Operations

ઑપરેશન	Syntax	વર્ણન
બનાવવું	tuple = (1,2,3)	નવું tuple બનાવો
એક્સેસ	tuple[0]	Index વડે element મેળવો
Count	tuple.count(2)	Occurrences ગિનો
Index	tuple.index(3)	પહેલો index શોધો
Slice	tuple[1:3]	Sub-tuple મેળવો
Length	len(tuple)	Tuple નું size મેળવો
Concatenate	tuple1 + tuple2	Tuples જોડો

Tuple Comparison Diagram:



આકૃતિ 4. Tuple vs List

કોડ ઉદાહરણ:

```
# Tuple creation and operations
coordinates = (10, 20, 30)
print(coordinates[0])  # 10
print(len(coordinates))  # 3
x, y, z = coordinates  # tuple unpacking
new_tuple = coordinates + (40, 50)
```

મેમરી ટ્રીક

``Ordered Immutable Heterogeneous Indexed''

પ્રશ્ન 3(અ) [3 ગુણ]

random મોડ્યૂલની કોઈ પણ 3 મેથડ સમજાવો.

જવાબ

કોષ્ટક 5. Random Module Methods

મેથડ	Syntax	વર્ણન
random()	random.random()	0.0 થી 1.0 વચ્ચે float
randint()	random.randint(a,b)	આપેલી range વચ્ચે integer
choice()	random.choice(list)	Sequence માંથી random element

કોડ ઉદાહરણ:

```python
import random
print(random.random())      # 0.7234567
print(random.randint(1, 10))  # 7
print(random.choice(['r','g','b'])) # g
```

મેમરી ટ્રીક

``Random Float, Randint Integer, Choice Select''

પ્રશ્ન 3(બ) [4 ગુણ]

વપરાશકર્તા પાસેથી એક સ્ટ્રિંગ લઈને એમાંના દરેક 'a' નું સ્થાન પ્રિન્ટ કરવાનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```python
# Get string from user
text = input("Enter a string: ")

# Find all positions of 'a'
positions = []
for i in range(len(text)):
    if text[i].lower() == 'a':
        positions.append(i)

# Display results
if positions:
    print(f"Letter 'a' found at positions: {positions}")
else:
    print("Letter 'a' not found in the string")
```

મુખ્ય મુદ્દાઓ:
- **Case-insensitive**: 'a' અને 'A' બંને શોધવા માટે .lower() ઉપયોગ કરો
- **Index tracking**: range અથવા enumerate ઉપયોગ કરો
- **આઉટપુટ ફોર્મેટ**: સ્પષ્ટ position indication

મેમરી ટ્રીક

``Loop Index Check Append Print''

પ્રશ્ન 3(ક) [7 ગુણ]

પાયથનનો string ડેટા ટાઈપ વિસ્તારથી સમજાવો.

જવાબ

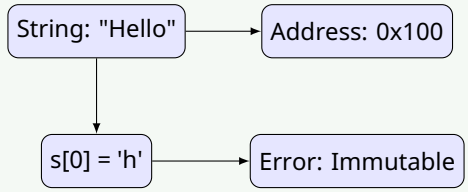
સ્ટ્રિંગની લાક્ષણિકતાઓ:
- **અપરિવર્તનીય**: બનાવ્યા પછી બદલી શકાતું નથી
- **અનુક્રમ**: Characters નો ordered collection
- **ઇન્ડેક્સવાળી**: Index વડે characters ને access કરી શકાય છે
- **યુનિકોડ**: બધી ભાષાઓ અને symbols સપોર્ટ કરે છે

સ્ટ્રિંગ મેથડ્સ ટેબલ:

કોષ્ટક 6. String Methods

મેથડ	ઉદાહરણ	વર્ણન
upper()	"s".upper()	Uppercase માં convert કરો
lower()	"S".lower()	Lowercase માં convert કરો
strip()	" s ".strip()	Whitespace દૂર કરો
split()	"a,b".split(",")	List માં split કરો
replace()	"s".replace("s","x")	Substring replace કરો
find()	"s".find("e")	Substring index શોધો

String Structure Diagram:



આકૃતિ 5. String Immutability

કોડ ઉદાહરણ:

```
name = "Python Programming"
print(name[0])     # P
print(name[0:6])   # Python
message = f"I love {name}"
```

મેમરી ટ્રીક

``Immutable Sequence Indexed Unicode''

પ્રશ્ન 3(અ OR) [3 ગુણ]

math મોડ્યૂલની કોઈ પણ 3 મેથડ સમજાવો.

જવાબ

કોષ્ટક 7. Math Module Methods

મેથડ	Syntax	વર્ણન
sqrt()	math.sqrt(16)	Square root ગણતરી
pow()	math.pow(2,3)	Power ગણતરી
ceil()	math.ceil(4.3)	Integer માં round up

કોડ ઉદાહરણ:

```
import math
print(math.sqrt(25))   # 5.0
print(math.pow(2, 3))  # 8.0
print(math.ceil(4.2))  # 5
```

મેમરી ટ્રીક

``Square Root, Power Up, Ceiling Round''

પ્રશ્ન 3(બ OR) [4 ગુણ]

વપરાશકર્તા પાસેથી એક સ્ટ્રિંગ લઈને એમાં રહેલા અંગ્રેજી સ્વરોની સંખ્યા શોધવાનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# Get string from user
text = input("Enter a string: ")

# Define vowels
vowels = "aeiouAEIOU"

# Count vowels
vowel_count = 0
for char in text:
    if char in vowels:
        vowel_count += 1

# Display result
print(f"Total vowels in '{text}': {vowel_count}")
```

મેમરી ટ્રીક

``Define Vowels, Loop Check, Count Increment''

પ્રશ્ન 3(ક OR) [7 ગુણ]

પાયથનનો set ડેટા ટાઈપ વિસ્તારથી સમજાવો.

જવાબ

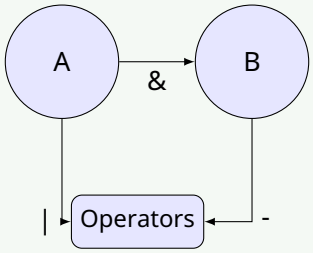
સેટની લાક્ષણિકતાઓ:
- **અક્રમ**: Elements નો કોઈ નિશ્ચિત sequence નથી
- **બદલાવ પાત્ર**: Elements ઉમેરી/દૂર કરી શકાય છે
- **અનન્ય**: Duplicate elements allowed નથી
- **પુનરાવર્તનીય**: Elements માં loop કરી શકાય છે

સેટ ઑપરેશન્સ ટેબલ:

કોષ્ટક 8. Set Operations

ઑપરેશન	Syntax	વર્ણન
બનાવવું	set = {1,2,3}	નવો set બનાવો
Add	set.add(4)	Single element ઉમેરો
Remove	set.remove(2)	Element દૂર કરો
સંયોજન	s1 | s2	Sets જોડો
છેદ	s1 & s2	સામાન્ય elements
તફાવત	s1 - s2	ફક્ત set1 માંના elements

Set Venn Diagram:



આકૃતિ 6. Set Logic

કોડ ઉદાહરણ:

```
A = {1, 2, 3, 4}
B = {3, 4, 5, 6}
print(A | B)   # Union: {1,2,3,4,5,6}
print(A & B)   # Intersection: {3,4}
```

મેમરી ટ્રીક

``Unordered Mutable Unique Iterable''

પ્રશ્ન 4(અ) [3 ગુણ]

પાયથનમાં ક્લાસ શું છે? તે ઓબ્જેક્ટથી કઈ રીતે અલગ છે?

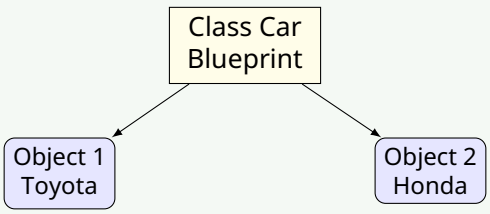
જવાબ

ક્લાસ વિ. ઓબ્જેક્ટ સરખામણી:

કોષ્ટક 9. Class vs Object

પાસું	ક્લાસ	ઓબ્જેક્ટ
વ્યાખ્યા	Blueprint અથવા template	ક્લાસનું instance
મેમરી	કોઈ memory allocate થતી નથી	Memory allocate થાય છે
અસ્તિત્વ	Logical entity	Physical entity
બનાવટ	class keyword ઉપયોગ કરીને	Class constructor ઉપયોગ કરીને

Relationship Diagram:



આકૃતિ 7. Class to Objects

મેમરી ટ્રીક

``Class Blueprint, Object Instance''

પ્રશ્ન 4(બ) [4 ગુણ]

dictionary ડેટા ટાઈપની કોઈ પણ 4 મેથડ સમજાવો.

જવાબ

Dictionary મેથડ્સ ટેબલ:

કોષ્ટક 10. Dictionary Methods

મેથડ	Syntax	વર્ણન
keys()	d.keys()	બધી keys મેળવો
values()	d.values()	બધી values મેળવો
items()	d.items()	Key-value pairs મેળવો
get()	d.get('k')	Value સુરક્ષિત રીતે મેળવો

કોડ ઉદાહરણ:

```
student = {'name': 'John', 'grade': 'A'}
print(student.keys())   # ['name', 'grade']
print(student.values())  # ['John', 'A']
print(student.get('age')) # None (no error)
```

મેમરી ટ્રીક

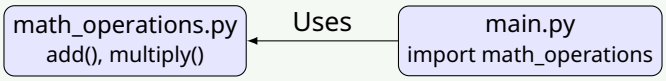
``Keys Values Items Get''

પ્રશ્ન 4(ક) [7 ગુણ]

કોઈ કાર્યો કરવા માટે યુઝર ડિફાઈન્ડ મોડ્યૂલ બનાવી તેને ઈમ્પોર્ટ કરી તેના ફંક્શનનો ઉપયોગ કરવાનો પાયથન પ્રોગ્રામ લખો.

જવાબ

Module Structure:



આકૃતિ 8. Module Import

મોડ્યૂલ બનાવવું (math_operations.py):

```
def add(a, b):
    return a + b

def multiply(a, b):
    return a * b

PI = 3.14159
```

મુખ્ય પ્રોગ્રામ (main.py):

```
import math_operations as mo

res1 = mo.add(5, 3)
res2 = mo.multiply(4, 6)
```

```
print(f"Addition: {res1}")
print(f"Multiplication: {res2}")
print(f"PI: {mo.PI}")
```

મુખ્ય મુદ્દાઓ:
- **મોડ્યૂલ બનાવવું:** Functions સાથે અલગ .py ફાઈલ
- **Import પદ્ધતિઓ:** import module અથવા from module import func
- **ઉપયોગ:** module.function()

મેમરી ટ્રીક

``Create Import Use''

પ્રશ્ન 4(અ OR) [3 ગુણ]

પાયથન ક્લાસની મેથડ્સના પ્રકારો ટૂંકમાં સમજાવો.

જવાબ

કોષ્ટક 11. Method Types

પ્રકાર	Decorator	First Argument
Instance	None	self
Class	@classmethod	cls
Static	@staticmethod	None

ઉદાહરણ:

```
class Demo:
    def inst(self): pass
    @classmethod
    def cls_m(cls): pass
    @staticmethod
    def stat(): pass
```

મેમરી ટ્રીક

``Instance Self, Class Cls, Static None''

પ્રશ્ન 4(બ OR) [4 ગુણ]

string ડેટા ટાઈપની કોઈ પણ 4 મેથડ સમજાવો.

જવાબ

String Methods: Checks Counts

કોષ્ટક 12. String Check Methods

મેથડ	Syntax	વર્ણન
startswith	s.startswith('a')	Substring થી શરૂ થાય છે કે ચેક કરો
endswith	s.endswith('z')	Substring થી અંત થાય છે કે ચેક કરો
isdigit	s.isdigit()	બધા digits છે કે ચેક કરો
count	s.count('a')	Substring ની occurrences ગિનો

કોડ ઉદાહરણ:

```
s = "Hello World 123"
print(s.startswith("He")) # True
print(s.endswith("23"))  # True
print("123".isdigit())   # True
print(s.count("l"))      # 3
```

મેમરી ટ્રીક

``Start End Digit Count''

પ્રશ્ન 4(ક OR) [7 ગુણ]

રિકર્સીવ ફંક્શનની મદદથી આપેલ નંબરનો ફેક્ટોરીયલ શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

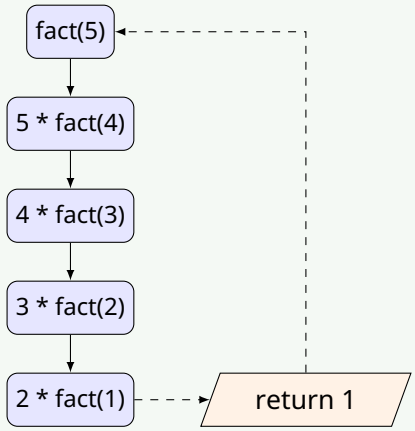
જવાબ

કોડ:

```
def factorial(n):
    # Base case
    if n == 0 or n == 1:
        return 1
    # Recursive case
    else:
        return n * factorial(n - 1)

try:
    num = int(input("Enter a number: "))
    if num < 0:
        print("Negative number not allowed")
    else:
        print(f"Factorial of {num} is {factorial(num)}")
except ValueError:
    print("Invalid input")
```

Recursion Stack Visualization:



આકૃતિ 9. Recursive Calls

મેમરી ટ્રીક

``Base Stop, Recursive Call, Error Check''

પ્રશ્ન 5(અ) [3 ગુણ]

સિંગલ ઇન્હેરિટન્સ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

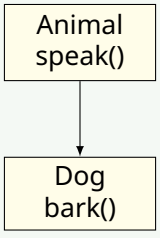
કોડ:

```
class Animal:
    def speak(self): print("Animal Speak")

class Dog(Animal):
    def bark(self): print("Dog Bark")

d = Dog()
d.speak()  # Inherited
d.bark()   # Own
```

Inheritance Diagram:



આકૃતિ 10. Single Inheritance

મેમરી ટ્રીક

``Parent Child Inherit Override''

પ્રશ્ન 5(બ) [4 ગુણ]

પાયથન ક્લાસમાં કન્સ્ટ્રક્ટરનું મહત્વ સમજાવો.

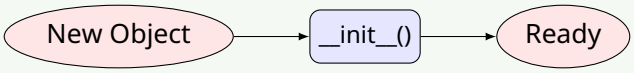
જવાબ

કન્સ્ટ્રક્ટરનું મહત્વ:

કોષ્ટક 13. Constructor Features

પાસું	વર્ણન
ઇનિશિયલાઇઝેશન	ઓબ્જેક્ટ બનાવવામાં આવે ત્યારે આપોઆપ call થાય છે
સેટઅપ	Instance variables ને values સાથે initialize કરે છે
મેમરી	Object attributes માટે memory allocate કરે છે
વેલિડેશન	Creation દરમિયાન input parameters validate કરે છે

Lifecycle Flow:



આકૃતિ 11. Constructor Flow

મેમરી ટ્રીક

``Initialize Setup Memory Validate''

પ્રશ્ન 5(ક) [7 ગુણ]

ઇન્હેરિટન્સ દ્વારા થતું મેથડ ઓવરરાઇડિંગ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

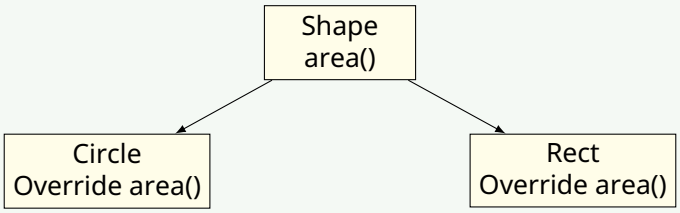
```
class Shape:
    def area(self): print("Shape Area")

class Circle(Shape):
    def area(self): print("Circle Area")

class Rect(Shape):
    def area(self): print("Rect Area")

shapes = [Circle(), Rect()]
for s in shapes:
    s.area()
```

Method Overriding Hierarchy:



આકૃતિ 12. Polymorphism

મુખ્ય મુદ્દાઓ:
- **સમાન મેથડ નામ**: Parent અને child classes માં
- **અલગ implementation**: Child class specific logic આપે છે
- **Runtime નિર્ણય**: Object type આધારે યોગ્ય method call થાય છે

મેમરી ટ્રીક

``Same Name Different Logic Runtime Decision''

પ્રશ્ન 5(અ OR) [3 ગુણ]

પાયથનમાં ડેટા એન્કેપ્સ્યુલેશનનો ખ્યાલ સમજાવો.

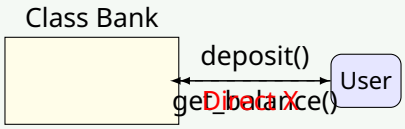
જવાબ

ડેટા એન્કેપ્સ્યુલેશન:

કોષ્ટક 14. Encapsulation

પાસું	વર્ણન
વ્યાખ્યા	Data અને methods ને એકસાથે બાંધવું
ડેટા છુપાવવું	Private attributes (__var)
એક્સેસ	Public methods (getters/setters)

Encapsulation Diagram:



આકૃતિ 13. Secure Access

મેમરી ટ્રીક

``Bundle Data Hide Interface''

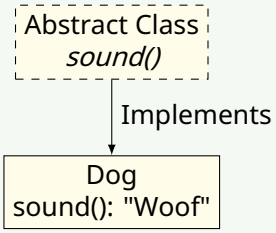
પ્રશ્ન 5(બ OR) [4 ગુણ]

પાયથનમાં એબ્સ્ટ્રેક્ટ ક્લાસનો ખ્યાલ સમજાવો.

જવાબ

એબ્સ્ટ્રેક્ટ ક્લાસ:
- **વ્યાખ્યા**: સીધા instantiate ન થઈ શકતો ક્લાસ
- **એબ્સ્ટ્રેક્ટ મેથડ્સ**: Declared પણ implemented નથી
- **અમલીકરણ**: Subclasses એ abstract methods implement કરવા જોઈએ
- **મોડ્યૂલ**: abc મોડ્યૂલ ઉપયોગ કરે છે

Abstraction Logic:



આકૃતિ 14. Abstract Base Class

મેમરી ટ્રીક

``Cannot Instantiate Force Implementation Common Interface''

પ્રશ્ન 5(ક OR) [7 ગુણ]

મલ્ટિપલ ઇન્હેરિટન્સ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

કોડ:

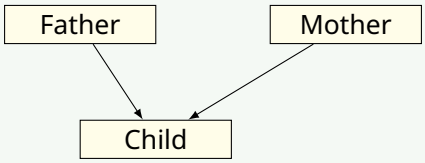
```
class Father:
    def work(self): print("Father Engineer")

class Mother:
    def work(self): print("Mother Doctor")

class Child(Father, Mother):
    pass

c = Child()
c.work() # Father (MRO)
print(Child.__mro__)
```

Multiple Inheritance Structure:



આકૃતિ 15. Multiple Parents

મુખ્ય મુદ્દાઓ:

- **Structure**: Child બંને Father અને Mother થી inherit કરે છે
- **MRO**: મેથડ રિઝોલ્યુશન ઓર્ડર નક્કી કરે છે
- **Diamond Problem**: C3 linearization વડે solve થાય છે

મેમરી ટ્રીક

``Multiple Parents MRO Constructor Diamond''