

# OOPS & Python Programming (4351108) - Winter 2023 Solution

Milav Dabgar

December 06, 2023

## પ્રશ્ન 1(અ) [3 ગુણ]

પાયથન પ્રોગ્રામિંગ લેન્જાવેજના કોઈ પણ 6 ઉપયોગો લખો.

### જવાબ

કોષ્ટક 1. પાયથનના ઉપયોગો

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

### મેમરી ટ્રીક

“Web Data Machine Desktop Game Auto”

## પ્રશ્ન 1(બ) [4 ગુણ]

પાયથન પ્રોગ્રામિંગ લેન્જાવેજની કોઈ પણ 8 વિશેષતાઓ લખો.

### જવાબ

કોષ્ટક 2. પાયથનની વિશેષતાઓ

વિશેષતા	વર્ણન
સરળ સિન્ટેક્સ	વાંચવા અને લખવામાં સરળ
ઇન્ટરપ્રિટેડ	Compilation ની જરૂર નથી
ઓફ્ઝ્લેક્ટ-ઓરિએન્ટેડ	OOP concepts સપોર્ટ કરે છે
ડાયનામિક ટાઇપિંગ	Variables ને type declaration જરૂરી નથી
કોસ-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''

## પ્રક્રિયા 1(ક) [૭ ગુણ]

પાયથનની for અને while લૂપનું કાર્ય સમજાવો.

## જવાબ

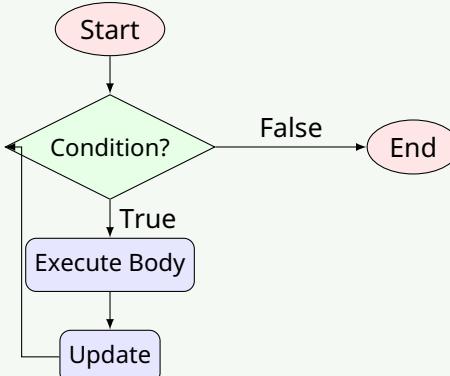
## For Loop:

- પુનરાવર્તન: Sequences પર પુનરાવર્તન કરે છે (lists, strings, ranges)
- સિટેક્સ: 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:

```

1 # For loop
2 for i in range(5):
3     print(i)
4
5 # While loop
6 i = 0
7 while i < 5:
8     print(i)
9     i += 1
  
```

## મેમરી ટ્રીક

``For Automatic, While Manual''

## પ્રક્રિયા 1(ક) OR) [૭ ગુણ]

પાયથનના break, continue અને pass સ્ટેમેન્ટના 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

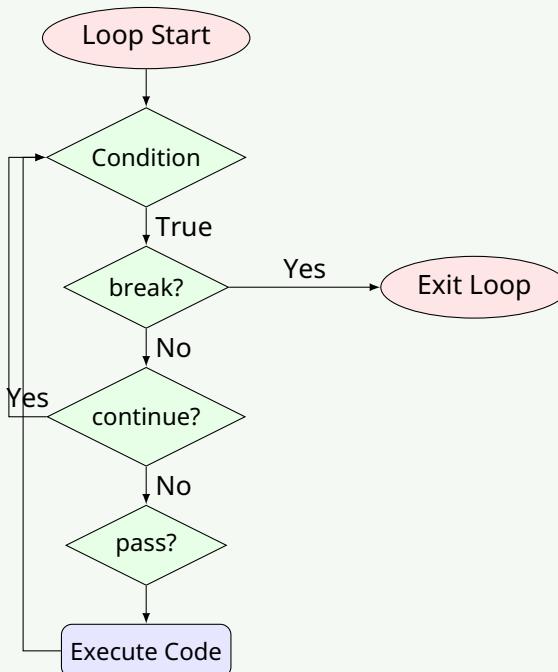
### Continue Statement:

- છોડીને આગળા: 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 નથી
- અસર: કોઈ operation perform કરતું નથી

### Control Flow Visualization:



આકૃતિ 2. Loop Control Statements

### કોડ ઉદાહરણો:

```

1 # Break
2 for i in range(10):
3     if i == 5: break
4     print(i) # 0,1,2,3,4
5
6 # Continue
7 for i in range(5):
8     if i == 2: continue
9     print(i) # 0,1,3,4
10
11 # Pass
12 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:

```

1 # Method 1 - Using for loop
2 numbers = [1, 2, 3, 4, 5]
3 for i in range(len(numbers)):
4     numbers[i] += 1
5 print(numbers)
6
7 # Method 2 - List comprehension
8 numbers = [1, 2, 3, 4, 5]
9 result = [x + 1 for x in numbers]
10 print(result)

```

### મેમરી ટ્રીક

“Loop Index or Comprehension”

## પ્રશ્ન 2(બ) [4 ગુણ]

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

### જવાબ

કોડ:

```

1 # Input three numbers
2 num1 = float(input("Enter first number: "))
3 num2 = float(input("Enter second number: "))
4 num3 = float(input("Enter third number: "))
5
6 # Calculate average
7 average = (num1 + num2 + num3) / 3
8
9 # Display result
10 print(f"Average is: {average}")

```

### મુખ્ય મુદ્દાઓ:

- ઇનપુટ: દર્શાંશ 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:

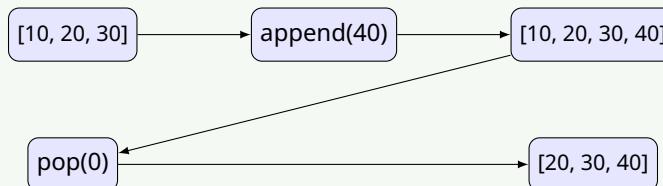
- કમ્પ્યુટરની જગતે છે
- બદલાવ પાત્ર: બનાવ્યા પછી modify કરી શકાય છે
- વિવિધ પ્રકારની: વિવિધ data types store કરી શકે છે
- ઇન્ડેક્સવાળી: Index વડે elements ને access કરી શકાય છે (0-based)

લિસ્ટ ઓપરેશન્સ ટેબલ:

કોષ્ટક 3. List Operations

ઓપરેશન	Syntax	વર્ણન
બનાવું	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:

```

1 # List creation and operations
2 fruits = ['apple', 'banana', 'orange']
3 fruits.append('mango')
4 fruits.insert(1, 'grape')
5 print(fruits[0]) # apple
6 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:

```

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

```

5     total += num
6     print(f"Sum is: {total}")
7
8 # Method 2 - Using range and index
9 numbers = [10, 20, 30, 40, 50]
10 total = 0
11 for i in range(len(numbers)):
12     total += numbers[i]
13     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 સાંદું વ્યાજ શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

## જવાબ

## કોડ:

```

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

5
6 # Calculate simple interest
7 simple_interest = (principal * rate * time) / 100

8
9 # Display results
10 print(f"Principal: {principal}")
11 print(f"Rate: {rate}%")
12 print(f"Time: {time} years")
13 print(f"Simple Interest: {simple_interest}")
14 print(f"Total Amount: {principal + simple_interest}")

```

## સૂત્ર:

- સાંદું વ્યાજ =  $(P \times R \times T) / 100$
- કુલ રકમ = મૂડલ + સાંદું વ્યાજ

## મેમરી ટ્રીક

``Principal Rate Time, Multiply Divide Hundred''

## પ્રશ્ન 2(ક) OR [7 ગુણ]

પાયથનનો tuple ડેટા ટાઈપ વિસ્તારથી સમજાવો.

## જવાબ

## ટ્યૂપલની લાક્ષણિકતાઓ:

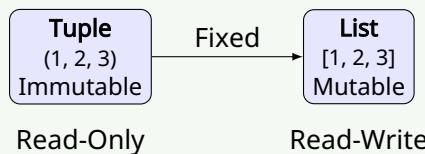
- ક્રમબદ્ધ: Elements sequence જાળવે છે

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

કોષ્ટક 4. Tuple Operations

ઓપરેશન	Syntax	વર્ણન
બનાવું	tuple = (1,2,3)	નવું tuple બનાવો
એક્સ્સેસ	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 ઉદાહરણાઃ

```

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

## મેમરી ટ્રીક

"Ordered Immutable Heterogeneous Indexed"

## પ્રશ્ન 3(અ) [3 ગુણ]

random મોડ્યુલની કોઈ પણ 3 મેથડ સમજાવો.

## જવાબ

કોષ્ટક 5. Random Module Methods

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

કોડ ઉદાહરણાઃ

```

1 import random
2 print(random.random())      # 0.7234567
3 print(random.randint(1, 10)) # 7
4 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:**

```

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

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

9 # Display results
10 if positions:
11     print(f"Letter 'a' found at positions: {positions}")
12 else:
13     print("Letter 'a' not found in the string")
14

```

**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:**

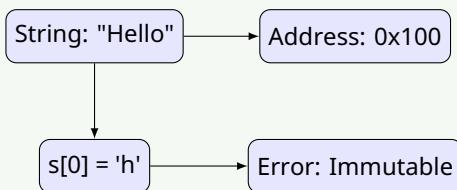
- અપરિવર્તનીય: બનાવ્યા પછી બદલી શકતું નથી
- અનુક્રમ: Characters નો ordered collection
- ઇન્ડેક્સવાળી: Index વડે characters ને access કરી શકાય છે
- ચુંભિકીડ: બધી ભાષાઓ અને symbols સપોર્ટ કરે છે

**સ્ટ્રિંગ મેથ્ડ્સ ટેબલ:**

### કોષ્ટક 6. String Methods

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

### String Structure Diagram:



### આકૃતિ 5. String Immutability

### કોડ ઉદાહરણ:

```

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

### મેમરી ટ્રીક

``Immutable Sequence Indexed Unicode''

## પ્રશ્ન 3(અ OR) [3 ગુણ]

math મોડ્યુલની કોઈ પણ 3 મેથડ સમજાવો.

### જવાબ

### કોષ્ટક 7. Math Module Methods

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

### કોડ ઉદાહરણ:

```

1 import math
2 print(math.sqrt(25))  # 5.0
3 print(math.pow(2, 3)) # 8.0
4 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 અંગેજુ સ્વરોની સંખ્યા શોધવાનો પાયથન પ્રોગ્રામ લખો.

## જવાબ

કોડ:

```

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

3
4 # Define vowels
5 vowels = "aeiouAEIOU"

6
7 # Count vowels
8 vowel_count = 0
9 for char in text:
10     if char in vowels:
11         vowel_count += 1

12
13 # Display result
14 print(f"Total vowels in '{text}': {vowel_count}")

```

## મેમરી ટ્રીક

"Define Vowels, Loop Check, Count Increment"

## પ્રશ્ન 3(ક) OR [7 ગુણ]

પાયથનનો set કેરા ટાઈપ વિસ્તારથી સમજાવો.

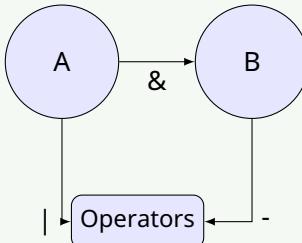
## જવાબ

સેટની લાક્ષણિકતાઓ:

- અક્રમ: Elements નો કોઈ નિશ્ચિયત sequence નથી
  - બદલાવ પાત્ર: Elements ઉમેરી/દૂર કરી શકાય છે
  - અન્યાય: Duplicate elements allowed નથી
  - પુનરાવર્ત્નીય: Elements માં loop કરી શકાય છે
- સેટ ઓપરેશન્સ ટેબલ:

કોષ્ટક 8. Set Operations

ઓપરેશન	Syntax	વર્ણન
બનાવું	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:**

```

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

**મેમરી ટ્રીક**

``Unordered Mutable Unique Iterable''

**પ્રક્રિયા 4(અ) [3 ગુણ]**

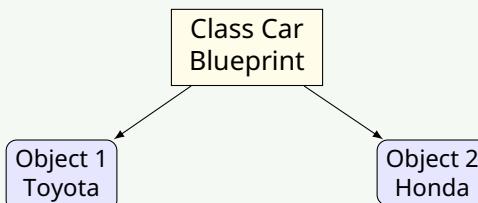
પાયથનમાં કલાસ શું છે? તે ઓફજેક્ટથી કઈ રીતે અલગ છે?

**જવાબ**

કલાસ વિ. ઓફજેક્ટ સરખામણી:

કોષ્ટક 9. Class vs Object

પાસું	કલાસ	ઓફજેક્ટ
વ્યાખ્યા	Blueprint અથવા template	કલાસનું 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''

## પ્રક્રિયા 4(બ) [4 ગુણ]

dictionary કેટા 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 ઉદાહરણાં:

```

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

```

### મેમરી ટ્રીક

“Keys Values Items Get”

## પ્રક્રિયા 4(ક) [7 ગુણ]

કોઈ કાર્યો કરવા માટે ચુંઝર ડિફાઈન્ડ મોડ્યુલ બનાવી તેને ઈંપોર્ટ કરી તેના ફંક્શનનો ઉપયોગ કરવાનો પાયથન પ્રોગ્રામ લખો.

### જવાબ

Module Structure:



આકૃતિ 8. Module Import

મોડ્યુલ બનાવવું (math\_operations.py):

```

1 def add(a, b):
2     return a + b
3
4 def multiply(a, b):
5     return a * b
6
7 PI = 3.14159

```

મુખ્ય પ્રોગ્રામ (main.py):

```

1 import math_operations as mo
2
3 res1 = mo.add(5, 3)
4 res2 = mo.multiply(4, 6)

```

```

5   print(f"Addition: {res1}")
6   print(f"Multiplication: {res2}")
7   print(f"PI: {mo.PI}")
8

```

**મુખ્ય મુદ્દાઓ:**

- મેડ્યુલ 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 કલાસની મેથડ્સના પ્રકારો ટૂકમાં સમજાવો.

**જવાબ****કોષ્ટક 11. Method Types**

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

**ઉદાહરણ:**

```

1 class Demo:
2     def inst(self): pass
3     @classmethod
4     def cls_m(cls): pass
5     @staticmethod
6     def stat(): pass

```

**મેમરી ટ્રીક****``Instance Self, Class Cls, Static None''****પ્રશ્ન 4(બ OR) [4 ગુણ]**

string ડેટા ટાઇપની કોઈ પણ 4 મેથડ સમજાવો.

**જવાબ****String Methods: Checks Counts****કોષ્ટક 12. String Check Methods**

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

કોડ ઉદાહરણ:

```

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

```

મેમરી ટ્રીક

``Start End Digit Count''

## પ્રશ્ન 4(ક) OR [7 ગુણ]

રિક્રીબ ફુંક્શનની મદદથી આપેલ નંબરનો ફેક્ટોરીયલ શોધવા માટેનો પાયથન પ્રોગ્રામ લખો.

જવાબ

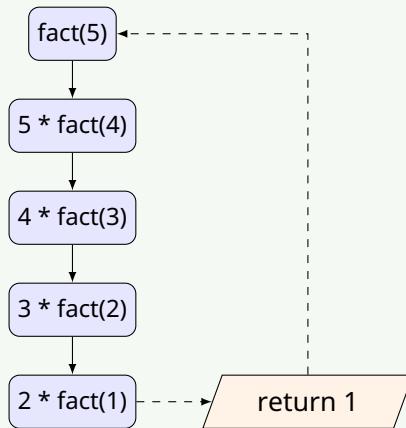
કોડ:

```

1 def factorial(n):
2     # Base case
3     if n == 0 or n == 1:
4         return 1
5     # Recursive case
6     else:
7         return n * factorial(n - 1)
8
9 try:
10    num = int(input("Enter a number: "))
11    if num < 0:
12        print("Negative number not allowed")
13    else:
14        print(f"Factorial of {num} is {factorial(num)}")
15    except ValueError:
16        print("Invalid input")

```

Recursion Stack Visualization:



આકૃતિ 9. Recursive Calls

## મેમરી ટ્રીક

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

## પ્રક્રિયા 5(અ) [3 ગુણ]

સિંગલ ઇનહેરિટન્સ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

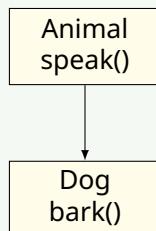
## જવાબ

## કોડ:

```

1 class Animal:
2     def speak(self): print("Animal Speak")
3
4 class Dog(Animal):
5     def bark(self): print("Dog Bark")
6
7 d = Dog()
8 d.speak() # Inherited
9 d.bark() # Own
  
```

## Inheritance Diagram:



આકૃતિ 10. Single Inheritance

## મેમરી ટ્રીક

``Parent Child Inherit Override''

## પ્રશ્ન 5(બ) [4 ગુણ]

પાયથન કલાસમાં કન્સ્ટ્રક્ટરનું મહત્વ સમજાવો.

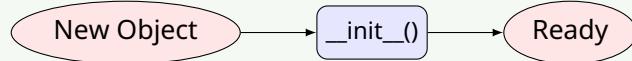
### જવાબ

કન્સ્ટ્રક્ટરનું મહત્વ:

કોષ્ટક 13. Constructor Features

પાસું	વર્ણન
ઇનિશિયલાઇઝરન	ઓફ્જેક્ટ 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]

ઇનહેરિટન્સ દ્વારા થતું મેથડ ઓવરરાઇડિંગ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

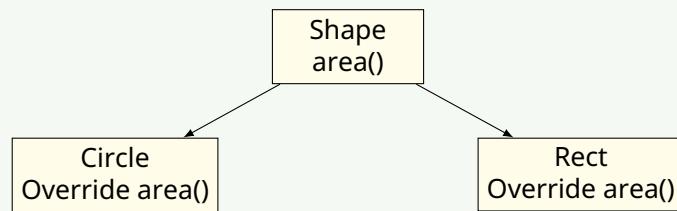
### જવાબ

કોડ:

```

1 class Shape:
2     def area(self): print("Shape Area")
3
4 class Circle(Shape):
5     def area(self): print("Circle Area")
6
7 class Rect(Shape):
8     def area(self): print("Rect Area")
9
10 shapes = [Circle(), Rect()]
11 for s in shapes:
12     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 એન્કેપ્ચ્યુલેશનનો ઘ્યાલ સમજાવો.

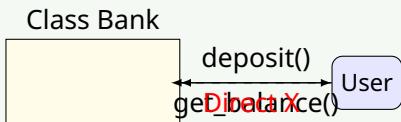
જવાબ

ડેટા એન્કેપ્ચ્યુલેશન:

## કોષ્ટક 14. Encapsulation

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

Encapsulation Diagram:



## આકૃતિ 13. Secure Access

મેમરી ટ્રીક

"Bundle Data Hide Interface"

## પ્રશ્ન 5(બ) OR [4 ગુણ]

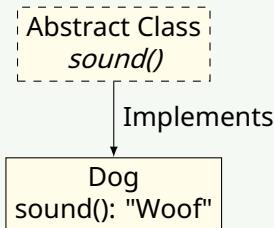
પાયથનમાં એબ્સ્ટ્રાક્ટ કલાસનો ઘ્યાલ સમજાવો.

જવાબ

એબ્સ્ટ્રાક્ટ કલાસ:

- વ્યાખ્યા: સીધા instantiate ન થઈ શકતો કલાસ
- એબ્સ્ટ્રાક્ટ મેથડ્સ: Declared પણ implemented નથી
- અમલીકરણ: Subclasses એ abstract methods implement કરવા જોઈએ
- મોડ્યુલ: 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]

મલિટિપલ ઇનહેરિટન્સ બતાવવા માટેનો પાયથન પ્રોગ્રામ લખો.

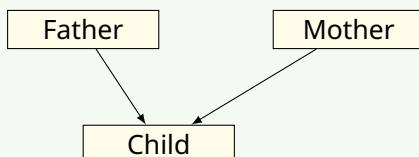
## જવાબ

## કોડ:

```

1 class Father:
2     def work(self): print("Father Engineer")
3
4 class Mother:
5     def work(self): print("Mother Doctor")
6
7 class Child(Father, Mother):
8     pass
9
10 c = Child()
11 c.work() # Father (MRO)
12 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''