OOPS & Python Programming (4351108) - Winter 2024 Solution

Milav Dabgar

November 25, 2024

પ્રશ્ન 1(અ) [3 ગુણ]

પાયથોન પ્રોગ્રામિંગ ભાષાના લક્ષણોની યાદી બનાવો.

જવાબ

કોષ્ટક 1. પાયથોનના લક્ષણો

લક્ષણ	વર્ણન
સરળ અને સહેલું	સ્વચ્છ, વાંચી શકાય તેવું syntax
મફત અને ઓપન સોર્સ	કોઈ કિંમત નહીં, community driven
ક્રોસ-પ્લેટફોર્મ	Windows, Linux, Mac પર ચાલે છે
Interpreted	compilation ની જરૂર નથી
Object-Oriented	classes અને objects ને support કરે છે
મોટી લાઇબ્રેરીઓ	સમૃદ્ધ standard library

મેમરી ટ્રીક

``Simple Free Cross Interpreted Object Large''

પ્રશ્ન 1(બ) [4 ગુણ]

પાયથોન પ્રોગ્રામિંગ ભાષાની એપ્લિકેશનો લખો.

જવાબ

કોષ્ટક 2. પાયથોન એપ્લિકેશનો

એપ્લિકેશન ક્ષેત્ર	ઉદાહરણો
વેબ ડેવલપમેન્ટ	Django, Flask frameworks
ડેટા સાયન્સ	NumPy, Pandas, Matplotlib
મશીન લર્નિંગ	TensorFlow, Scikit-learn
ડેસ્કટોપ GUI	Tkinter, PyQt applications
ગેમ ડેવલપમેન્ટ	Pygame library
ઓટોમેશન	Scripting અને testing

મેમરી ટ્રીક

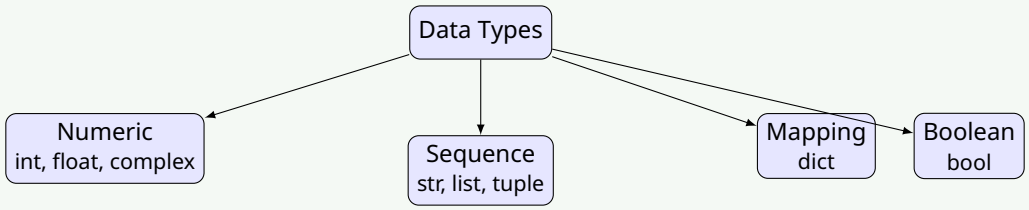
``Web Data Machine Desktop Game Auto''

પ્રશ્ન 1(ક) [7 ગુણ]

પાયથોનમાં વિવિધ ડેટાટાઇપ્સ સમજાવો.

જવાબ

Data Type Hierarchy:



આકૃતિ 1. Python Data Types

કોષ્ટક 3. Python Data Types

ડેટા ટાઇપ	ઉદાહરણ	વર્ણન
int	x = 5	પૂર્ણાંક સંખ્યાઓ
float	y = 3.14	દશાંશ સંખ્યાઓ
str	name = "John"	ટેક્સ્ટ ડેટા
bool	flag = True	True/False મૂલ્યો
list	[1, 2, 3]	ક્રમબદ્ધ, બદલી શકાય તેવું
tuple	(1, 2, 3)	ક્રમબદ્ધ, બદલી ન શકાય તેવું
dict	{"a": 1}	Key-value જોડી
set	{1, 2, 3}	અનન્ય ઘટકો

કોડ ઉદાહરણ:

```
# Numeric types
age = 25        # int
price = 99.99    # float

# Text type
name = "Python"   # str

# Boolean type
is_valid = True   # bool

# Collection types
numbers = [1, 2, 3]       # list
coordinates = (10, 20)    # tuple
student = {"name": "John"} # dict
unique_ids = {1, 2, 3}    # set
```

મેમરી ટ્રીક

``Integer Float String Boolean List Tuple Dict Set''

પ્રશ્ન 1(ક OR) [7 ગુણ]

એરિથમેટિક, એસાઇનમેન્ટ અને આઇડેન્ટિટી ઓપરેટરો ઉદાહરણ સાથે સમજાવો.

જવાબ

એરિથમેટિક ઓપરેટરો:

કોષ્ટક 4. એરિથમેટિક ઓપરેટરો

Op	Name	ઉદાહરણ
+	બાકીદારી	5 + 3 = 8
-	બાદબાકી	5 - 3 = 2
*	ગુણાકાર	5 * 3 = 15
/	ભાગાકાર	10 / 3 = 3.33
//	Floor Div	10 // 3 = 3
%	Modulus	10 % 3 = 1
**	ઘાત	2 ** 3 = 8

એસાઇનમેન્ટ ઓપરેટરો:

કોષ્ટક 5. એસાઇનમેન્ટ ઓપરેટરો

Op	ઉદાહરણ	સમકક્ષ
=	x = 5	મૂલ્ય આપો
+=	x += 3	x = x + 3
-=	x -= 2	x = x - 2
*=	x *= 4	x = x * 4

આઇડેન્ટિટી ઓપરેટરો:

કોષ્ટક 6. આઇડેન્ટિટી ઓપરેટરો

Op	હેતુ	ઉદાહરણ
is	સમાન ઓબ્જેક્ટ	x is y
is not	વિવિધ ઓબ્જેક્ટ	x is not y

કોડ ઉદાહરણ:

```
# Arithmetic
a = 10 + 5   # 15
b = 10 // 3  # 3

# Assignment
x = 5
x += 3       # x બને છે 8

# Identity
list1 = [1, 2, 3]
list2 = [1, 2, 3]
print(list1 is list2)     # False
print(list1 is not list2) # True
```

મેમરી ટ્રીક

``Add Assign Identity''

પ્રશ્ન 2(અ) [3 ગુણ]

નીચેનામાંથી કયા આઇડેન્ટિફાયર્સ ના નામો અમાન્ય છે? (i) Total Marks (ii) Total_Marks (iii) total-Marks (iv)

Hundred$ (v) _Percentage (vi) True

જવાબ

કોષ્ટક 7. Identifier Validity

આઇડેન્ટિફાયર	માન્ય/અમાન્ય	કારણ
Total Marks	**અમાન્ય**	સ્પેસ છે
Total_Marks	માન્ય	અન્ડરસ્કોર મંજૂર છે
total-Marks	**અમાન્ય**	હાઇફન મંજૂર નથી
Hundred$	**અમાન્ય**	$ સિમ્બોલ મંજૂર નથી
_Percentage	માન્ય	અન્ડરસ્કોરથી શરૂ થઈ શકે છે
True	**અમાન્ય**	આરક્ષિત કીવર્ડ છે

અમાન્ય આઇડેન્ટિફાયર્સ: Total Marks, total-Marks, Hundred$, True

મેમરી ટ્રીક

``Space Hyphen Dollar Keyword = Invalid''

પ્રશ્ન 2(બ) [4 ગુણ]

આપેલ ત્રણ સંખ્યાઓમાંથી મહત્તમ સંખ્યા શોધવા માટે પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# ત્રણ સંખ્યાઓ input લો
num1 = float(input("પ્રથમ સંખ્યા દાખલ કરો: "))
num2 = float(input("બીજી સંખ્યા દાખલ કરો: "))
num3 = float(input("ત્રીજી સંખ્યા દાખલ કરો: "))

# if-elif-else વાપરીને મહત્તમ શોધો
if num1 >= num2 and num1 >= num3:
    maximum = num1
elif num2 >= num1 and num2 >= num3:
    maximum = num2
else:
    maximum = num3

# પરિણામ દર્શાવો
print(f"મહત્તમ સંખ્યા છે: {maximum}")
```

Alternative using max() function:

```
num1, num2, num3 = map(float, input("3 સંખ્યાઓ દાખલ કરો: ").split())
maximum = max(num1, num2, num3)
print(f"મહત્તમ: {maximum}")
```

મેમરી ટ્રીક

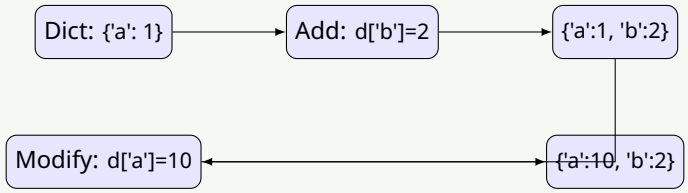
``Input Compare Display''

પ્રશ્ન 2(ક) [7 ગુણ]

પાયથોનમાં ડિક્શનરી સમજાવો. ડિક્શનરીમાં ઘટકો ઉમેરવા, બદલવા અને કાઢી નાખવા માટેના સ્ટેટમેન્ટ લખો.

જવાબ

ડિક્શનરી: એક collection જે ક્રમબદ્ધ, બદલાય તેવો અને ડુપ્લિકેટ keys નથી મંજૂર કરે છે.
Dictionary Operations:



આકૃતિ 2. Dictionary Operations

કોષ્ટક 8. Dictionary Methods

ઓપરેશન	સિન્ટેક્સ	ઉદાહરણ
બનાવો	dict = {}	student = {}
ઉમેરો	d[k] = v	student['name'] = 'John'
બદલો	d[k] = new	student['name'] = 'Jane'
ડિલીટ	del d[k]	del student['name']
એક્સેસ	d[k]	print(student['name'])

કોડ ઉદાહરણ:

```
# ખાલી ડક્ષ્શનરી બનાવો
student = {}

# ઘટકો ઉમેરો
student['name'] = 'John'
student['age'] = 20

# ઘટક બદલો
student['age'] = 21

# ઘટક ડલિટ કરો
del student['name']

# ડક્ષ્શનરી દર્શાવો
print(student)  # આઉટપુટ: {'age': 21}
```

મેમરી ટ્રીક

``Key-Value Ordered Changeable Unique''

પ્રશ્ન 2(અ OR) [3 ગુણ]

નીચેની પેટર્ન દર્શાવવા માટેનો પ્રોગ્રામ લખો.

જવાબ

પેટર્ન:

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

કોડ:

```
# પેટર્ન પ્રોગ્રામ
for i in range(1, 6):
    for j in range(1, i + 1):
        print(j, end=" ")
    print()  # દરેક રો પછી નવી લાઇન
```

મેમરી ટ્રીક

``Outer Row Inner Column Print''

પ્રશ્ન 2(બ OR) [4 ગુણ]

વપરાશકર્તા દ્વારા દાખલ કરેલ પૂર્ણાંક સંખ્યાના અંકોનો સરવાળો શોધવા માટે પ્રોગ્રામ લખો.

જવાબ

કોડ:

```
# વપરાશકર્તા પાસેથી સંખ્યા input લો
number = int(input("સંખ્યા દાખલ કરો: "))
original_number = number
sum_digits = 0

# અંકો કાઢો અને સરવાળો કરો
while number > 0:
    digit = number % 10    # છેલ્લો અંક મેળવો
    sum_digits += digit    # સરવાળામાં ઉમેરો
    number = number // 10  # છેલ્લો અંક દૂર કરો

# પરિણામ દર્શાવો
print(f"{original_number} ના અંકોનો સરવાળો છે: {sum_digits}")
```

વૈકલ્પિક રીત:

```
number = input("સંખ્યા દાખલ કરો: ")
sum_digits = sum(int(digit) for digit in number)
print(f"અંકોનો સરવાળો: {sum_digits}")
```

મેમરી ટ્રીક

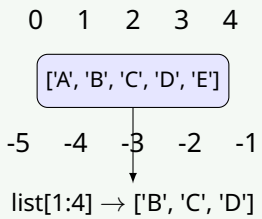
``Input Extract Sum Display''

પ્રશ્ન 2(ક OR) [7 ગુણ]

લિસ્ટમાં સ્લાઇસિંગ અને કન્કેટનેશન ઓપરેશન સમજાવો.

જવાબ

લિસ્ટ સ્લાઇસિંગ: [start:stop:step] સિન્ટેક્સ વાપરીને લિસ્ટનો ભાગ કાઢવો.
Slicing Visualization:



આકૃતિ 3. List Indexing & Slicing

ઓપરેશન્સ ટેબલ:

કોષ્ટક 9. લિસ્ટ ઓપરેશન્સ

સિન્ટેક્સ	વર્ણન	ઉદાહરણ
l[start:stop]	start થી stop-1 સુધી	nums[1:4]
l[:stop]	શરૂઆતથી	nums[:3]
l[::step]	step સાથે	nums[::2]
l1 + l2	કન્કેટનેશન	[1]+[2]

કોડ ઉદાહરણ:

```
list1 = [1, 2, 3, 4, 5]
list2 = [6, 7, 8]

# સ્લાઇસિંગ
print(list1[1:4])   # [2, 3, 4]
print(list1[::-1])  # [5, 4, 3, 2, 1]

# કન્કેટનેશન
result = list1 + list2  # [1, 2, 3... 8]
list1.extend(list2)     # list1 ને modify કરે છે
```

મેમરી ટ્રીક

``Slice Extract Concat Join''

પ્રશ્ન 3(અ) [3 ગુણ]

પાયથોનમાં લિસ્ટ વ્યાખ્યાયિત કરો. લિસ્ટના અંતમાં એલિમેન્ટ ઉમેરવા માટે વપરાતા ફંક્શનનું નામ લખો.

જવાબ

લિસ્ટ: એક ક્રમબદ્ધ, બદલાય તેવો સંગ્રહ.
ગુણધર્મો:
- **ક્રમબદ્ધ**: આઇટમ્સનો નિશ્ચિત ક્રમ છે
- **બદલાય તેવું**: બનાવ્યા પછી બદલી શકાય છે
- **ડુપ્લિકેટ્સ**: ડુપ્લિકેટ મૂલ્યો મંજૂર કરે છે

ઘટક ઉમેરવા માટેનું ફંક્શન: append()
ઉદાહરણ:

```
fruits = ['apple', 'banana']
fruits.append('orange')
print(fruits)  # ['apple', 'banana', 'orange']
```

મેમરી ટ્રીક

``List Append End''

પ્રશ્ન 3(બ) [4 ગુણ]

પાયથોનમાં ટ્યુપલ વ્યાખ્યાયિત કરો. ટ્યુપલના છેલ્લા એલિમેન્ટને એક્સેસ કરવા માટેનું સ્ટેટમેન્ટ લખો.

જવાબ

ટ્યુપલ: એક ક્રમબદ્ધ, બદલાય તેવો ન હોય તેવો સંગ્રહ.
છેલ્લા એલિમેન્ટને એક્સેસ કરવું:

```
my_tuple = (10, 20, 30, 40, 50)

# મેથડ 1: નેગેટિવ ઇન્ડેક્સ
last = my_tuple[-1]  # 50

# મેથડ 2: Length
last = my_tuple[len(my_tuple) - 1] # 50
```

મેમરી ટ્રીક

``Tuple Unchangeable Negative Index''

પ્રશ્ન 3(ક) [7 ગુણ]

નીચેના સેટ ઓપરેશન્સ માટે સ્ટેટમેન્ટ લખો: ખાલી સેટ બનાવો, સેટમાં એક ઘટક ઉમેરો, સેટમાંથી એક ઘટક દૂર કરો, બે સેટનું યુનિયન, બે સેટનું છેદ, બે સેટ વચ્ચેનો તફાવત અને બે સેટ વચ્ચે સિમેટ્રિક તફાવત.

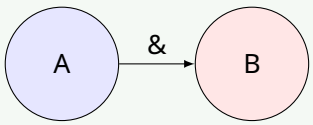
જવાબ

સેટ ઓપરેશન્સ ટેબલ:

કોષ્ટક 10. સેટ ઓપરેશન્સ

ઓપરેશન	મેથડ/Op	ઉદાહરણ
ખાલી બનાવો	set()	s = set()
ઉમેરો	add()	s.add(5)
દૂર કરો	remove()	s.remove(5)
યુનિયન	union() |	A | B
છેદ (Intersect)	intersection() &	A & B
તફાવત (Diff)	difference() -	A - B
સિમેટ્રિક તફાવત	sym_diff() ^	A ^ B

Set Relations Diagram:



Union: All Intersect: Common

આકૃતિ 4. Set Relations

કોડ ઉદાહરણ:

```
s = set()
s.add(10)
s.remove(10)

A = {1, 2, 3}
B = {3, 4, 5}
print(A | B)  # {1, 2, 3, 4, 5}
print(A & B)  # {3}
print(A - B)  # {1, 2}
print(A ^ B)  # {1, 2, 4, 5}
```

મેમરી ટ્રીક

``Create Add Remove Union Intersect Differ Symmetric''

પ્રશ્ન 3(અ OR) [3 ગુણ]

પાયથોનમાં સ્ટ્રિંગ વ્યાખ્યાયિત કરો. ઉદાહરણ વાપરીને સમજાવો (i) સ્ટ્રિંગ કેવી રીતે બનાવવી. (ii) ઇન્ડેક્સિંગનો ઉપયોગ કરીને વ્યક્તિગત અક્ષરોને એક્સેસ કરવું.

જવાબ

સ્ટ્રિંગ: અવતરણચિહ્ન (સિંગલ, ડબલ) માં બંધ કરેલા અક્ષરોનો ક્રમ.
String Structure:

0	1	2	3	4	5
P	Y	T	H	O	N
-6	-5	-4	-3	-2	-1

આકૃતિ 5. String Indexing

કોડ:

```
# બનાવટ
s1 = 'Hello'
s2 = "World"

# એક્સેસિંગ
word = "PYTHON"
print(word[0])  # P
print(word[-1])  # N
```

મેમરી ટ્રીક

``String Quotes Index Access''

પ્રશ્ન 3(બ OR) [4 ગુણ]

ફોર લૂપ અને વ્હાઇલ લૂપનો ઉપયોગ કરીને લિસ્ટ ટ્રાવર્સિંગ સમજાવો.

જવાબ

List Traversing: લિસ્ટના દરેક ઘટકને મુલાકાત લેવી.
Comparison:

કોષ્ટક 11. Loop Comparison

ફોર લૂપ	વ્હાઇલ લૂપ
સરળ સિન્ટેક્સ	વધુ નિયંત્રણ
Iterations ખબર હોય ત્યારે	શરત આધારિત માટે

કોડ ઉદાહરણ:

```
nums = [10, 20, 30]

# For Loop
for x in nums:
    print(x)

# While Loop
i = 0
while i < len(nums):
    print(nums[i])
    i += 1
```

મેમરી ટ્રીક

``For Simple While Control''

પ્રશ્ન 3(ક OR) [7 ગુણ]

એક એવો પ્રોગ્રામ લખો કે જેનાથી વર્ગમાં n વિદ્યાર્થીઓના રોલ નંબર, નામ અને માર્ક્સ સાથેની ડિક્શનરી બનાવી શકાય અને 75 થી વધુ ગુણ મેળવનારા વિદ્યાર્થીઓના નામ ડિસ્પ્લે કરી શકાય.

જવાબ

કોડ:

```
# વિદ્યાર્થીઓની સંખ્યા input કરો
n = int(input("વિદ્યાર્થીઓની સંખ્યા દાખલ કરો: "))
students = {}

# ડેટા input કરો
for i in range(n):
    print(f"\nStudent {i + 1}:")
    roll = int(input("Roll: "))
    name = input("Name: ")
    marks = float(input("Marks: "))

    students[roll] = {'name': name, 'marks': marks}

# 75 થી વધુ માર્ક્સ વાળા
print("\nStudents with marks > 75:")
found = False
for roll, data in students.items():
    if data['marks'] > 75:
        print(f"Name: {data['name']}, Marks: {data['marks']}")
        found = True

if not found:
    print("None found")
```

મેમરી ટ્રીક

``Input Store Filter Display''

પ્રશ્ન 4(અ) [3 ગુણ]

રેન્ડમ મોડ્યુલમાં ઉપલબ્ધ કોઈપણ ત્રણ ફંક્શન લખો. દરેક ફંક્શનનું સિન્ટેક્સ અને ઉદાહરણ લખો.

જવાબ

કોષ્ટક 12. Random Functions

ફંક્શન	વર્ણન	ઉદાહરણ
random()	0.0 થી 1.0 Float	0.75
randint(a,b)	a થી b Integer	5
choice(seq)	Random element	'red'

કોડ:

```
import random
print(random.random())
print(random.randint(1, 10))
print(random.choice(['a', 'b', 'c']))
```

મેમરી ટ્રીક

``Random Randint Choice''

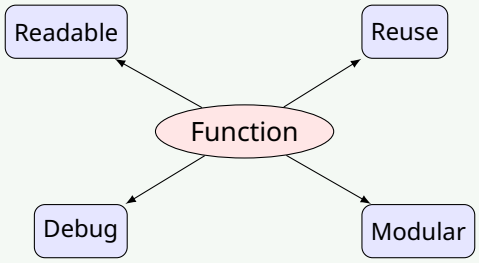
પ્રશ્ન 4(બ) [4 ગુણ]

ફંક્શનના ફાયદા લખો.

જવાબ

ફાયદા:
- **Code Reusability**: કોડ પુનઃઉપયોગ.
- **Modularity**: પ્રોગ્રામને નાના ભાગોમાં વહેંચવો.
- **Debugging**: એરર્સ શોધવી સરળ છે.
- **Readability**: સંગઠિત કોડ.

Concept Map:



આકૃતિ 6. Function Advantages

મેમરી ટ્રીક

``Reuse Modular Debug Read Maintain Avoid''

પ્રશ્ન 4(ક) [7 ગુણ]

એક પ્રોગ્રામ લખો જે વપરાશકર્તાને સ્ટ્રિંગ માટે પૂછે અને સ્ટ્રિંગમાં દરેક 'a' નું સ્થાન પ્રિન્ટ કરે.

જવાબ

કોડ:

```
text = input("સ્ટ્રિંગ દાખલ કરો: ")
positions = []

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

# Display
if positions:
    print(f"'a' found at indices: {positions}")
    for pos in positions:
        print(f"Index {pos}: '{text[pos]}'")
else:
    print("'a' not found")
```

મેમરી ટ્રીક

``Input Loop Check Store Display''

પ્રશ્ન 4(અ OR) [3 ગુણ]

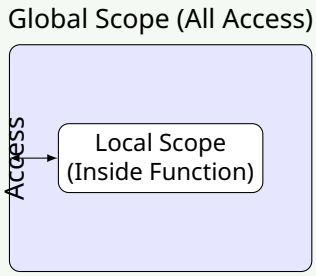
લોકલ અને ગ્લોબલ વેરિયેબલ સમજાવો.

જવાબ

કોષ્ટક 13. Variable Scopes

પ્રકાર	સ્કોપ	એક્સેસ
Local	ફંક્શનની અંદર	ફંક્શનમાં જ
Global	આખા પ્રોગ્રામમાં	બધે જ

Scope Visualization:



આકૃતિ 7. Variable Scope

કોડ:

```
g = 10  # Global

def func():
    l = 5    # Local
    print(g) # Access Global
```

મેમરી ટ્રીક

``Local Inside Global Everywhere''

પ્રશ્ન 4(બ OR) [4 ગુણ]

યુઝર ડિફાઇન્ડ ફંક્શનનું બનાવટ અને ઉપયોગ ઉદાહરણ સાથે સમજાવો.

જવાબ

સિન્ટેક્સ:

```
def function_name(params):
    """Docstring"""
    # Body
    return value
```

ઘટકો: 1. **def**: કીવર્ડ 2. **Name**: નામ 3. **Parameters**: ઇનપુટ 4. **Return**: આઉટપુટ
ઉદાહરણ:

```
def greet(name):
    return f"Hello {name}"

```

```
msg = greet("John")
print(msg)
```

મેમરી ટ્રીક

``Define Call Return Parameter''

પ્રશ્ન 4(ક OR) [7 ગુણ]

calcFact() નામનું યુઝર ડિફાઇન્ડ ફંક્શન બનાવવા માટેનો પ્રોગ્રામ લખો કે જે આર્ગ્યુમેન્ટ તરીકે આપેલ સંખ્યાના ફેક્ટોરિયલની ગણતરી કરી તેને ડિસ્પ્લે કરે.

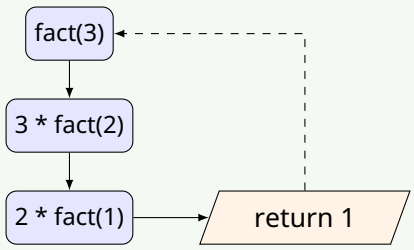
જવાબ

કોડ:

```
def calcFact(n):
    if n < 0:
        return "Undefined"
    elif n == 0 or n == 1:
        return 1
    else:
        fact = 1
        for i in range(2, n + 1):
            fact *= i
        return fact

# Logic
num = int(input("સંખ્યા દાખલ કરો: "))
print(f"Factorial of {num} is {calcFact(num)}")
```

Recursive Visual:



આકૃતિ 8. Recursion Stack

મેમરી ટ્રીક

``Define Check Loop Multiply Return''

પ્રશ્ન 5(અ) [3 ગુણ]

ક્લાસ અને ઓબ્જેક્ટ વચ્ચે તફાવત આપો.

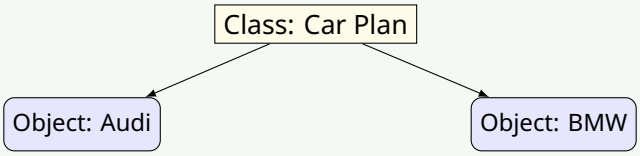
જવાબ

તફાવત:

કોષ્ટક 14. Class vs Object

બાબત	Class	Object
વ્યાખ્યા	બ્લૂપ્રિન્ટ	ઇન્સ્ટન્સ
મેમરી	એલોકેટ નથી	એલોકેટ છે
બનાવટ	class કીવર્ડ	કન્સ્ટ્રક્ટર કૉલ

Diagram:



આકૃતિ 9. Blueprint vs Instances

મેમરી ટ્રીક

``Class Blueprint Object Instance''

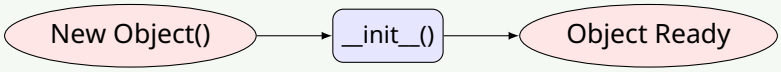
પ્રશ્ન 5(બ) [4 ગુણ]

ક્લાસમાં કન્સ્ટ્રક્ટરનો હેતુ જણાવો.

જવાબ

હેતુ:
- **Initialize**: ઓબ્જેક્ટ ઇનિશિયલાઇઝ કરો.
- **Automatic**: આપોઆપ કૉલ થાય છે.
- **Memory**: મેમરી એલોકેટ કરે છે.

Lifecycle:



આકૃતિ 10. Constructor Flow

મેમરી ટ્રીક

``Initialize Automatic Memory Default''

પ્રશ્ન 5(ક) [7 ગુણ]

"Student" નામનો ક્લાસ બનાવવા માટે પ્રોગ્રામ લખો જેમાં નામ, રોલ નંબર અને માર્ક્સ જેવા એટ્રિબ્યુટ્સ હોય. વિદ્યાર્થીની માહિતી પ્રદર્શિત કરવાની મેથડ બનાવો. "Student" ક્લાસનો ઓબ્જેક્ટ બનાવો અને મેથડનો ઉપયોગ કેવી રીતે કરવો તે બતાવો.

જવાબ

કોડ:

```
class Student:
    def __init__(self, name, roll, marks):
        self.name = name
        self.roll = roll
        self.marks = marks

    def display_info(self):
        print("-" * 20)
        print(f"Name: {self.name}")
        print(f"Roll: {self.roll}")
        print(f"Marks: {self.marks}")
        print("-" * 20)

# Objects બનાવો
s1 = Student("John", 101, 85)
s2 = Student("Alice", 102, 90)

# મેથડ વાપરો
s1.display_info()
s2.display_info()
```

મેમરી ટ્રીક

``Class Attributes Constructor Methods Objects''

પ્રશ્ન 5(અ OR) [3 ગુણ]

એન્કેપ્સ્યુલેશનનો હેતુ જણાવો.

જવાબ

Encapsulation: ડેટા અને મેથડ્સને સાથે રાખવું અને ડેટાને સુરક્ષિત કરવું.
હેતુ:
- **Data Hiding**: ડેટા છુપાવવો.
- **Security**: સુરક્ષા.
- **Controlled Access**: નિયંત્રિત એક્સેસ.

કોડ:

```
class Bank:
    def __init__(self):
        self.__bal = 0  # Private

    def deposit(self, amt):
        self.__bal += amt
```

મેમરી ટ્રીક

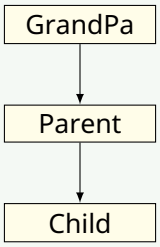
``Hide Protect Control Secure Modular''

પ્રશ્ન 5(બ OR) [4 ગુણ]

મલ્ટિલેવલ ઇન્હેરિટન્સ સમજાવો.

જવાબ

વ્યાખ્યા: ઇન્હેરિટન્સની શૃંખલા (A ← B ← C).
Diagram:



આકૃતિ 11. Multilevel Inheritance

કોડ:

```
class A: pass
class B(A): pass
class C(B): pass
obj = C() # A, B, C ના features છે
```

મેમરી ટ્રીક

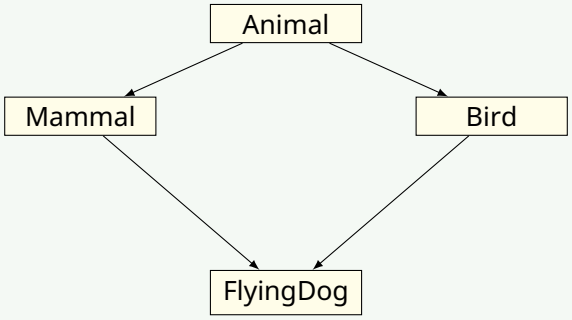
``Chain Inherit Level Access''

પ્રશ્ન 5(ક OR) [7 ગુણ]

હાઇબ્રિડ ઇન્હેરિટન્સનું કાર્ય દર્શાવતો પાયથોન પ્રોગ્રામ લખો.

જવાબ

Hybrid Inheritance: વિવિધ ઇન્હેરિટન્સનું સંયોજન.
Diagram:



આકૃતિ 12. Hybrid Inheritance

કોડ:

```
class Animal:
    def __init__(self): print("Animal")

```

```
class Mammal(Animal):
    def feed(self): print("Milk")

class Bird(Animal):
    def fly(self): print("Flying")

class FlyingDog(Mammal, Bird):
    def bark(self): print("Bark")

# Object
fd = FlyingDog()
fd.feed()  # Mammal
fd.fly()   # Bird
fd.bark()  # Own
```

મેમરી ટ્રીક

``Hybrid Multiple Single Multilevel Combined''