OOPS & Python Programming (4351108) - Summer 2024 Solution

Milav Dabgar

May 18, 2024

પ્રશ્ન 1(a) [3 ગુણ]

Python માં for loop નું કાર્ય સમજાવો.

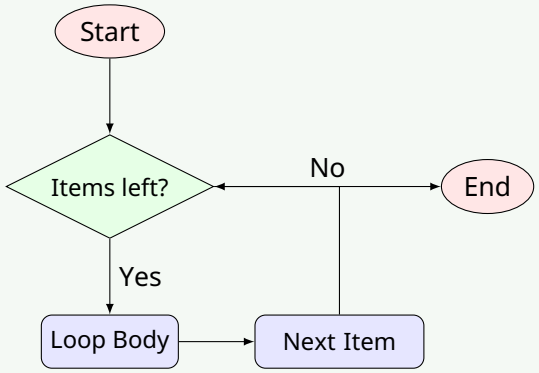
જવાબ

For loop એ list, tuple અથવા string જેવા sequence ના દરેક item માટે code block ને repeat કરે છે.
સિન્ટેક્સ ટેબલ:

કોષ્ટક 1. For Loop Syntax

ઘટક	Syntax	ઉદાહરણ
મૂળભૂત	for variable in sequence:	for i in [1,2,3]:
Range	for i in range(n):	for i in range(5):
String	for char in string:	for c in "hello":

આકૃતિ:



આકૃતિ 1. For Loop Execution Flow

- **પુનરાવર્તન:** Loop variable ને sequence માંથી દરેક value એક પછી એક મળે છે
- **આપમેળે:** Python આપમેળે next item પર જવાનું handle કરે છે
- **લવચીક:** Lists, strings, tuples, ranges સાથે કામ કરે છે

મેમરી ટ્રીક

``દરેક Item માટે, Block Execute કરો''

પ્રશ્ન 1(b) [4 ગુણ]

Python માં if-elif-else નું કાર્ય સમજાવો.

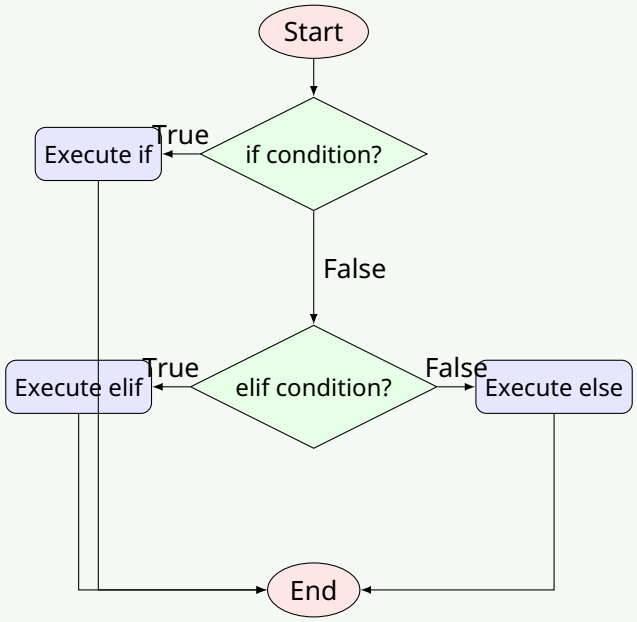
જવાબ

બહુ-માર્ગીય નિર્ણય માળખું જે sequence માં અનેક conditions ને ચકાસે છે.
માળખાકીય ટેબલ:

કોષ્ટક 2. If-Elif-Else Structure

Statement	હેતુ	Syntax
if	પ્રથમ શરત	if condition1:
elif	વૈકલ્પિક શરતો	elif condition2:
else	મૂળભૂત કેસ	else:

પ્રવાહ આકૃતિ:



આકૃતિ 2. If-Elif-Else Logic Flow

- **ક્રમબદ્ધ:** ઉપરથી નીચે conditions ને ચકાસે છે
- **વિશિષ્ટ:** માત્ર એક જ block execute થાય છે
- **વૈકલ્પિક:** elif અને else વૈકલ્પિક છે

મેમરી ટ્રીક

``જો આ, અથવા જો તે, અથવા Default''

પ્રશ્ન 1(c) [7 ગુણ]

Python પ્રોગ્રામનું માળખું સમજાવો.

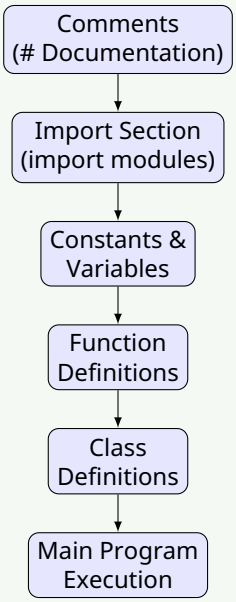
જવાબ

Python પ્રોગ્રામમાં તાર્કિક ક્રમમાં વિશિષ્ટ ઘટકો સાથે વ્યવસ્થિત માળખું હોય છે.
પ્રોગ્રામ માળખું ટેબલ:

કોષ્ટક 3. Python Program Structure

ઘટક	હેતુ	ઉદાહરણ
Comments	દસ્તાવેજીકરણ	# This is comment
Import	બાહ્ય modules	import math
Constants	નિશ્ચિત વેલ્યુઝ	PI = 3.14
Functions	પુનઃઉપયોગી કોડ	def function_name():
Classes	Objects નો blueprint	class ClassName:
Main code	પ્રોગ્રામ execution	if __name__ == "__main__":

પ્રોગ્રામ આર્કિટેક્ચર:



આકૃતિ 3. Python Program Structure

- **મોડ્યુલર**: દરેક વિભાગનો વિશિષ્ટ હેતુ હોય છે
- **વાંચવા યોગ્ય**: સ્પષ્ટ સંગઠન સમજવામાં મદદ કરે છે
- **જાળવણી યોગ્ય**: ફેરફાર અને debug કરવું સરળ
- **માનક**: Python conventions ને અનુસરે છે

સરળ ઉદાહરણ:

```
# Program to calculate area
import math

PI = 3.14159

def calculate_area(radius):
    return PI * radius * radius

# Main execution
radius = float(input("Enter radius: "))
area = calculate_area(radius)
print(f"Area = {area}")
```

મેમરી ટ્રીક

``Comment, Import, Constant, Function, Class, Main''

પ્રશ્ન 1(c OR) [7 ગુણ]

Python પ્રોગ્રામિંગ લેંગવેજની વિશેષતાઓ સમજાવો.

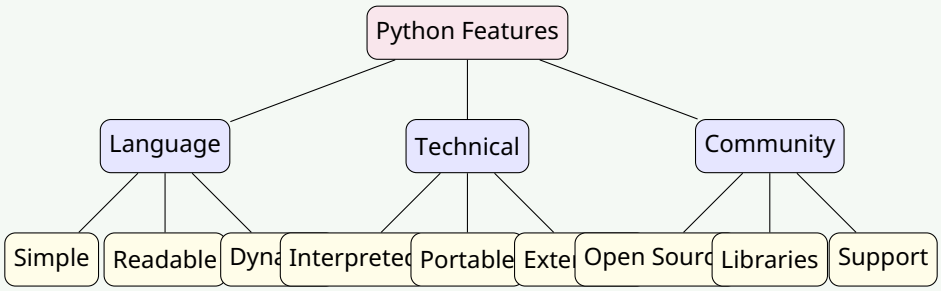
જવાબ

Python ની અનન્ય લાક્ષણિકતાઓ છે જે તેને beginners અને professionals માટે લોકપ્રિય બનાવે છે.
Python વિશેષતાઓ ટેબલ:

કોષ્ટક 4. Python Features

વિશેષતા	વર્ણન	લાભ
સરળ	સરળ syntax	ઝડપી શીખવા
Interpreted	કોઈ compilation નહીં	ઝડપી development
Object-Oriented	Classes અને objects	કોડની પુનઃઉપયોગીતા
Open Source	ઉપયોગ માટે મફત	કોઈ licensing ખર્ચ નહીં
Cross-Platform	દરેક જગ્યાએ run થાય છે	ઉચ્ચ portability

વિશેષતા કેટેગરીઝ:



આકૃતિ 4. Python Features Hierarchy

- **શિખાઉ-મિત્ર:** અંગ્રેજી ભાષા જેવું સરળ syntax
- **બહુમુખી:** web, AI, data science, automation માટે ઉપયોગ
- **સમૃદ્ધ લાયબ્રેરીઝ:** પ્રી-બિલ્ટ modules નો વિશાળ સંગ્રહ
- **ડાયનેમિક ટાઇપિંગ:** variable types declare કરવાની જરૂર નથી

કોડ ઉદાહરણ:

```
# Simple Python syntax
name = "Python"
print(f"Hello, {name}!")
```

મેમરી ટ્રીક

``સરળ, Interpreted, Object-Oriented, Open, Cross-platform''

પ્રશ્ન 2(a) [3 ગુણ]

સ્ટ્રિંગ પર થતાં કોઈ 3 ઓપરેશન સમજાવો.

જવાબ

String operations વિવિધ રીતે text data ને manipulate અને process કરે છે.
સ્ટ્રિંગ ઓપરેશન્સ ટેબલ:

કોષ્ટક 5. String Operations

ઓપરેશન	Method	ઉદાહરણ	પરિણામ
જોડવું	+	"Hello" + "World"	"HelloWorld"
લંબાઈ	len()	len("Python")	6
મોટા અક્ષર	.upper()	"hello".upper()	"HELLO"

ઓપરેશન ઉદાહરણો:

```
text = "Python"
# 1. જોડવું
result1 = text + " Programming"
# 2. લંબાઈ શોધવી
result2 = len(text)
# 3. મોટા અક્ષરમાં કન્વર્ટ કરવું
result3 = text.upper()
```

મેમરી ટ્રીક

``જોડો, ગણો, કન્વર્ટ કરો''

પ્રશ્ન 2(b) [4 ગુણ]

તાપમાનને ફેરનહાઇટથી સેલ્સિયસ એકમમાં (C=(F-32)/1.8 સમીકરણથી) પરિવર્તિત કરવા માટેનો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ user input સાથે ગાણિતિક formula વાપરીને temperature convert કરે છે.
એલ્ગોરિધમ ટેબલ:

કોષ્ટક 6. Conversion Algorithm

પગલું	ક્રિયા	કોડ
1	Input લો	fahrenheit = float(input())
2	Formula લાગુ કરો	celsius = (fahrenheit - 32) / 1.8
3	પરિણામ દર્શાવો	print(f"Celsius: {celsius}")

સંપૂર્ણ પ્રોગ્રામ:

```
# Temperature conversion program
fahrenheit = float(input("Enter temperature in Fahrenheit: "))
celsius = (fahrenheit - 32) / 1.8
print(f"Temperature in Celsius: {celsius:.2f}")
```

ટેસ્ટ કેસેસ:
- Input: 32°F → Output: 0.00°C
- Input: 100°F → Output: 37.78°C

મેમરી ટ્રીક

``Input, Calculate, Output''

પ્રશ્ન 2(c) [7 ગુણ]

Python માં list ડેટા ટાઇપ વિસ્તૃત રીતે સમજાવો.

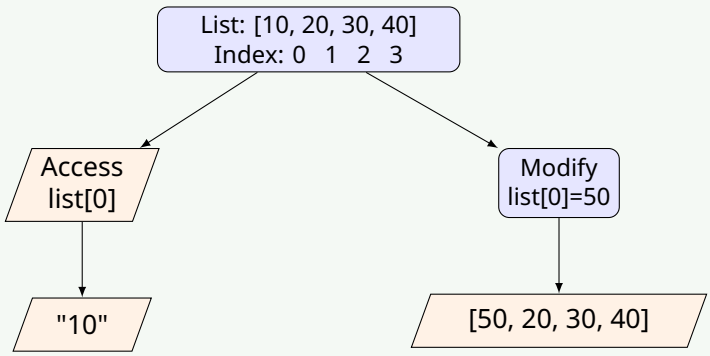
જવાબ

List એ ordered, mutable collection છે જે single variable માં multiple items store કરે છે.
લિસ્ટ લાક્ષણિકતાઓ ટેબલ:

કોષ્ટક 7. List Characteristics

પ્રોપર્ટી	વર્ણન	ઉદાહરણ
ક્રમબદ્ધ	Items નો position હોય છે	[1, 2, 3]
પરિવર્તનશીલ	બદલાઈ શકાય છે	list[0] = 10
ઇન્ડેક્સ્ડ	Position દ્વારા access	list[0]
મિશ્ર પ્રકારો	વિવિધ data types	[1, "hello", 3.14]

લિસ્ટ ઓપરેશન્સ આકૃતિ:



આકૃતિ 5. List Operations

સામાન્ય લિસ્ટ મેથડ્સ:
- append(): અંતે item ઉમેરો
- insert(): position પર ઉમેરો
- remove(): item ડિલીટ કરો
- pop(): છેલ્લું item દૂર કરો

ઉદાહરણ કોડ:

```
# Creating and using lists
numbers = [1, 2, 3, 4, 5]
numbers.append(6)      # અંતે 6 ઉમેરો
numbers.insert(0, 0)   # શરૂઆતમાં 0 ઉમેરો
print(numbers[2])      # જુઓ3 element access કરો
numbers.remove(3)      # value 3 દૂર કરો
```

મેમરી ટ્રીક

``ક્રમબદ્ધ, પરિવર્તનશીલ, ઇન્ડેક્સ્ડ, મિશ્ર''

પ્રશ્ન 2(a OR) [3 ગુણ]

Python માં સ્ટ્રિંગ ફોર્મેટિંગ સમજાવો.

જવાબ

String formatting એ templates માં values insert કરીને formatted strings બનાવે છે.
ફોર્મેટિંગ મેથડ્સ ટેબલ:

કોષ્ટક 8. Formatting Methods

Method	Syntax	ઉદાહરણ
f-strings	f"text {variable}"	f"Hello {name}"
format()	"text {}".format(value)	"Age: {}".format(25)
% operator	"text %s" % value	"Name: %s" % "John"

ઉપયોગ ઉદાહરણ:

```
name = "Alice"
age = 25
# f-string formatting
message = f"Hello {name}, you are {age} years old"
```

મેમરી ટ્રીક

``Format, Insert, Display''

પ્રશ્ન 2(b OR) [4 ગુણ]

સ્કેન કરેલ નંબર એકી સંખ્યા છે કે બેકી સંખ્યા છે તે ઓળખી અને યોગ્ય મેસેજ પ્રિન્ટ કરતો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ number 2 થી divisible છે કે નહીં તે ચકાસીને even અથવા odd નક્કી કરે છે.
લૉજિક ટેબલ:

કોષ્ટક 9. Even/Odd Logic

શરત	પરિણામ	મેસેજ
number % 2 == 0	Even	"Number is even"
number % 2 != 0	Odd	"Number is odd"

સંપૂર્ણ પ્રોગ્રામ:

```
# Even/Odd checker program
number = int(input("Enter a number: "))
if number % 2 == 0:
    print(f"{number} is even")
else:
    print(f"{number} is odd")
```

ટેસ્ટ કેસેસ:
- Input: 4 → Output: "4 is even"
- Input: 7 → Output: "7 is odd"

મેમરી ટ્રીક

``Input, Check Remainder, Display Result''

પ્રશ્ન 2(c OR) [7 ગુણ]

Python માં Set ડેટા ટાઇપ વિસ્તૃત રીતે સમજાવો.

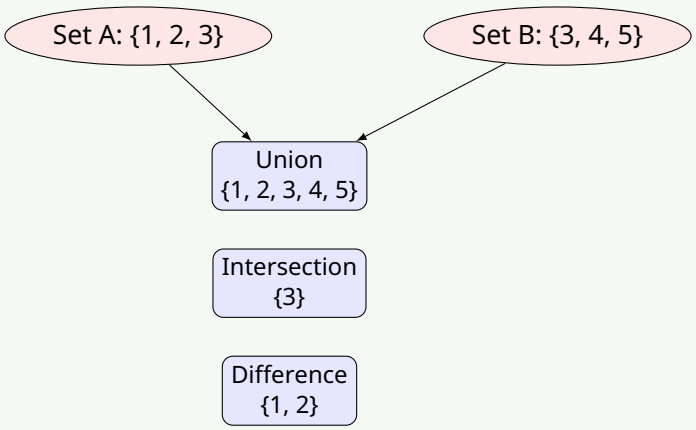
જવાબ

Set એ unordered collection છે જેમાં unique items હોય છે અને duplicate values નહીં.
સેટ લાક્ષણિકતાઓ ટેબલ:

કોષ્ટક 10. Set Characteristics

પ્રોપર્ટી	વર્ણન	ઉદાહરણ
અક્રમ	કોઈ નિશ્ચિત position નથી	{1, 3, 2}
અનન્ય	કોઈ duplicates નથી	{1, 2, 3}
પરિવર્તનશીલ	ફેરફાર કરી શકાય	set.add(4)
પુનરાવર્તન યોગ્ય	Loop કરી શકાય	for item in set:

સેટ ઓપરેશન્સ આકૃતિ:



આકૃતિ 6. Set Operations

સેટ મેથડ્સ ટેબલ:

કોષ્ટક 11. Set Methods

Method	હેતુ	ઉદાહરણ
add()	single item ઉમેરો	set.add(6)
update()	multiple items ઉમેરો	set.update([7, 8])
remove()	item ડિલીટ કરો	set.remove(3)
union()	sets જોડો	set1.union(set2)
intersection()	સામાન્ય items	set1.intersection(set2)

ઉદાહરણ કોડ:

```
# Creating and using sets
fruits = {"apple", "banana", "orange"}
fruits.add("mango")           # single item ઉમેરો
fruits.update(["grape", "kiwi"]) # multiple ઉમેરો
fruits.remove("banana")       # item દૂર કરો
print(len(fruits))            # items ગણો
```

મેમરી ટ્રીક

``અનન્ય, અક્રમ, પરિવર્તનશીલ, ગાણિતિક''

પ્રશ્ન 3(a) [3 ગુણ]

math મૉડ્યુલની કોઈ પણ 3 મેથડ સમજાવો.

જવાબ

Math module જટિલ ગણતરીઓ માટે ગાણિતિક functions પ્રદાન કરે છે.
મેથ મેથડ્સ ટેબલ:

કોષ્ટક 12. Math Methods

Method	હેતુ	ઉદાહરણ	પરિણામ
math.sqrt()	વર્ગમૂળ	math.sqrt(16)	4.0
math.pow()	પાવર ગણતરી	math.pow(2, 3)	8.0
math.ceil()	ઉપર રાઉન્ડ	math.ceil(4.3)	5

ઉપયોગ ઉદાહરણ:

```
import math
number = 16
result1 = math.sqrt(number)  # વર્ગમૂળ
result2 = math.pow(2, 4)    # 2 ની પાવર 4
result3 = math.ceil(7.2)    # 8 સુધી રાઉન્ડ અપ
```

મેમરી ટ્રીક

``વર્ગમૂળ, પાવર, સીલિંગ''

પ્રશ્ન 3(b) [4 ગુણ]

for loop નો ઉપયોગ કરીને લિસ્ટમાં આવેલ તમામ ઘટકોનો સરવાળો શોધવા માટેનો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ list દ્વારા iterate કરે છે અને બધા elements નો sum accumulate કરે છે.
એલ્ગોરિધમ ટેબલ:

કોષ્ટક 13. Summation Algorithm

પગલું	ક્રિયા	કોડ
1	Sum initialize કરો	total = 0
2	List માં loop કરો	for element in list:
3	Sum માં ઉમેરો	total += element
4	પરિણામ દર્શાવો	print(total)

સંપૂર્ણ પ્રોગ્રામ:

```
# Sum of list elements
numbers = [10, 20, 30, 40, 50]
```

```
total = 0
for element in numbers:
    total += element
print(f"Sum of all elements: {total}")
```

ટેસ્ટ કેસ:
- Input: [1, 2, 3, 4, 5] → Output: 15

મેમરી ટ્રીક

``Initialize, Loop, Add, Display''

પ્રશ્ન 3(c) [7 ગુણ]

બે list ની લંબાઈ સમાન છે કે નહીં તે ચકાસવા, અને જો હોય તો તેમને ભેગા કરીને તેમાંથી એક dictionary બનાવવાનો Python પ્રોગ્રામ વિકસાવો.

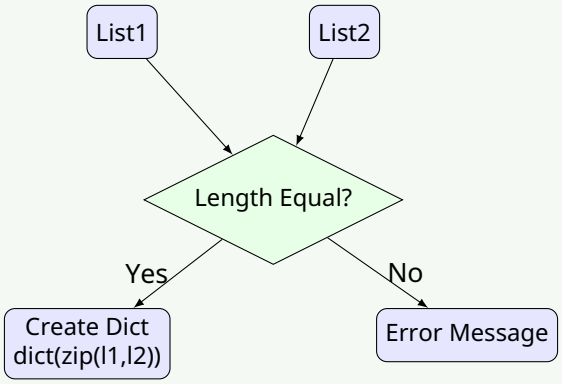
જવાબ

પ્રોગ્રામ list lengths ની સરખામણી કરે છે અને જો તે match કરે તો dictionary બનાવે છે.
લૉજિક ફ્લો ટેબલ:

કોષ્ટક 14. Merge Logic

પગલું	શરત	ક્રિયા
1	લંબાઈ ચકાસો	len(list1) == len(list2)
2	જો સમાન	Merge અને dictionary બનાવો
3	જો અસમાન	Error message દર્શાવો

પ્રક્રિયા આકૃતિ:



આકૃતિ 7. List Merge Logic

સંપૂર્ણ પ્રોગ્રામ:

```
# Merge lists into dictionary
list1 = ['name', 'age', 'city']
list2 = ['John', 25, 'Mumbai']

if len(list1) == len(list2):
    # Create dictionary using zip
    result_dict = dict(zip(list1, list2))
    print("Dictionary created:", result_dict)
else:
```

```
    print("Lists have different lengths, cannot merge")
```

અપેક્ષિત આઉટપુટ:

```
Dictionary created: {'name': 'John', 'age': 25, 'city': 'Mumbai'}
```

મેમરી ટ્રીક

``લંબાઈ ચકાસો, Zip કરો, Dictionary બનાવો''

પ્રશ્ન 3(a OR) [3 ગુણ]

statistics મૉડ્યુલની કોઈ પણ 3 મેથડ સમજાવો.

જવાબ

Statistics module numeric data પર statistical calculations માટે functions પ્રદાન કરે છે.
સ્ટેટિસ્ટિક્સ મેથડ્સ ટેબલ:

કોષ્ટક 15. Statistics Methods

Method	હેતુ	ઉદાહરણ	પરિણામ
statistics.mean()	સરેરાશ value	mean([1,2,3,4,5])	3.0
statistics.median()	મધ્ય value	median([1,2,3,4,5])	3
statistics.mode()	સૌથી વધુ વારંવાર	mode([1,1,2,3])	1

ઉપયોગ ઉદાહરણ:

```
import statistics
data = [10, 20, 30, 40, 50]
avg = statistics.mean(data)     # સરેરાશ કેલ્ક્યુલેટ કરો
mid = statistics.median(data)   # મધ્ય value શોધો
```

મેમરી ટ્રીક

``Mean, Median, Mode''

પ્રશ્ન 3(c OR) [7 ગુણ]

આપેલ સ્ટ્રિંગમાં કોઈ અક્ષર કેટલી વાર આવે છે તે ગણવા માટેની dictionary બનાવવાનો Python પ્રોગ્રામ વિકસાવો.

મેમરી ટ્રીક

``Loop, Check, Count, Store''

પ્રશ્ન 4(a) [3 ગુણ]

Python ક્લાસ અને ઓબ્જેક્ટ્સનું કાર્ય ઉદાહરણ સાથે સમજાવો.

મેમરી ટ્રીક

``ક્લાસ Blueprint, ઓબ્જેક્ટ Instance''

પ્રશ્ન 4(b) [4 ગુણ]

લિસ્ટમાં આવેલી તમામ એકી સંખ્યાઓ પ્રિન્ટ કરવા માટેનો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ list elements ને filter કરે છે અને માત્ર odd numbers દર્શાવે છે.
એકી સંખ્યા ચકાસણી ટેબલ:

કોષ્ટક 18. Odd Number Logic

સંખ્યા	Mod 2 (mod)	પરિણામ
1	1	એકી
2	0	બેકી

સંપૂર્ણ પ્રોગ્રામ:

```
# Print odd numbers from list
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print("Odd numbers in the list:")
for number in numbers:
    if number % 2 != 0:
        print(number, end=" ")
```

અપેક્ષિત આઉટપુટ:

```
Odd numbers in the list:
1 3 5 7 9
```

મેમરી ટ્રીક

``Loop, Check Remainder, Print Odd''

પ્રશ્ન 4(c) [7 ગુણ]

Python માં યુઝર ડિફાઇન્ડ ફંક્શન્સનું કાર્ય સમજાવો.

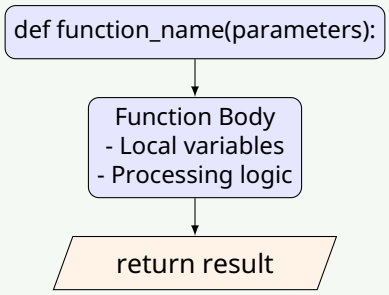
જવાબ

User-defined functions એ programmers દ્વારા બનાવેલા custom functions છે જે વિશિષ્ટ કાર્યો કરે છે.
ફંક્શન ઘટકો ટેબલ:

કોષ્ટક 19. Function Components

ઘટક	હેતુ	Syntax
def કીવર્ડ	Function declaration	def function_name():
Parameters	Input values	def func(param1, param2):
Body	Function code	Indented statements
return	Output value	return value

ફંક્શન માળખું:



આકૃતિ 10. Function Anatomy

ફંક્શન પ્રકારો:
- કોઈ parameters નહીં: def greet():
- Parameters સાથે: def add(a, b):
- Return value: return a + b
- કોઈ return નહીં: print("Hello")

ઉદાહરણ ફંક્શન્સ:

```
# Function with parameters and return value
def calculate_area(length, width):
    area = length * width
    return area

# Using functions
result = calculate_area(5, 3)
print(f"Area: {result}")
```

મેમરી ટ્રીક

``Define, Parameters, Body, Return''

પ્રશ્ન 4(a OR) [3 ગુણ]

Python માં કન્સ્ટ્રક્ટરનું કાર્ય સમજાવો.

જવાબ

Constructor એ special method છે જે objects બનાવવામાં આવે ત્યારે તેમને initialize કરે છે.
કન્સ્ટ્રક્ટર વિગતો ટેબલ:

કોષ્ટક 20. Constructor Details

પાસું	વર્ણન	Syntax
Method name	હંમેશા __init__	def __init__(self):
હેતુ	Object initialize કરવું	Initial values set કરવા
આપમેળે કૉલ	Object creation દરમ્યાન કૉલ થાય	obj = Class()

કન્સ્ટ્રક્ટર ઉદાહરણ:

```
class Student:
    def __init__(self, name, age):
        self.name = name
        self.age = age
        print("Student object created")

# Object creation automatically calls constructor
student1 = Student("Alice", 20)
```

- **આપમેળે એક્ઝિક્યુશન**: Object બનાવાતી વખતે તરત જ run થાય છે
- **ઇનિશિયલાઇઝેશન**: Object ની શરૂઆતી state set કરે છે
- **self પેરામીટર**: હાલનો object જે બનાવાઈ રહ્યો છે તેનો reference

મેમરી ટ્રીક

``Initialize, Automatic, Self''

પ્રશ્ન 4(b OR) [4 ગુણ]

min ફંક્શનનો ઉપયોગ કર્યા વિના લિસ્ટમાંથી સૌથી નાનો નંબર શોધવા માટેનો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ manually બધા elements ની સરખામણી કરીને સૌથી નાની value શોધે છે.
મિનિમમ શોધવાનો એલ્ગોરિધમ:

કોષ્ટક 21. Min Finding Algorithm

પગલું	ક્રિયા	કોડ
1	પહેલું smallest માનો	smallest = list[0]
2	બીજાઓ સાથે સરખાવો	for num in list[1:]:
3	નાનું મળે તો અપડેટ કરો	if num < smallest:
4	પરિણામ દર્શાવો	print(smallest)

સંપૂર્ણ પ્રોગ્રામ:

```
# Find smallest number without min()
numbers = [45, 23, 67, 12, 89, 5, 34]

smallest = numbers[0]  # પ્રથમને smallest માનો

for i in range(1, len(numbers)):
    if numbers[i] < smallest:
        smallest = numbers[i]

print(f"Smallest number: {smallest}")
```

અપેક્ષિત આઉટપુટ:

```
Smallest number: 5
```

મેમરી ટ્રીક

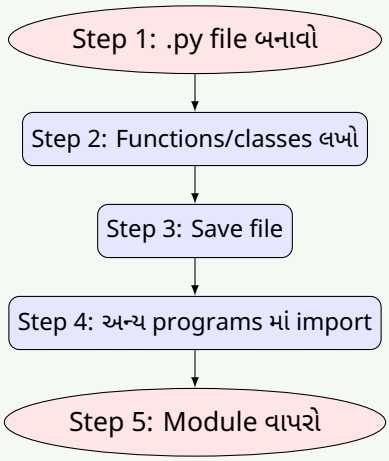
``માનો, સરખાવો, અપડેટ કરો, દર્શાવો''

પ્રશ્ન 4(c OR) [7 ગુણ]

Python માં યુઝર ડિફાઇન્ડ મોડ્યુલ્સનું કાર્ય સમજાવો.

જવાબ

User-defined modules એ custom Python files છે જેમાં functions, classes અને variables હોય છે જે અન્ય programs માં import અને use કરી શકાય છે.
મોડ્યુલ ઘટકો: Functions, Classes, Variables, Constants.
મોડ્યુલ બનાવવાની પ્રક્રિયા:



આકૃતિ 11. Module Lifecycle

ઉદાહરણ મોડ્યુલ (math_operations.py):

```python
PI = 3.14159

def calculate_circle_area(radius):
    return PI * radius * radius
```

મુખ્ય પ્રોગ્રામ:

```python
import math_operations

# Module functions વાપરવા
radius = 5
area = math_operations.calculate_circle_area(radius)
print(f"Circle area: {area}")
```

મોડ્યુલ લાભો:
- **કોડ પુનઃઉપયોગીતા**: એકવાર લખો, અનેક programs માં વાપરો
- **સંગઠન**: સંબંધિત functions એકસાથે રાખો
- **નેમસ્પેસ**: Naming conflicts ટાળો

મેમરી ટ્રીક

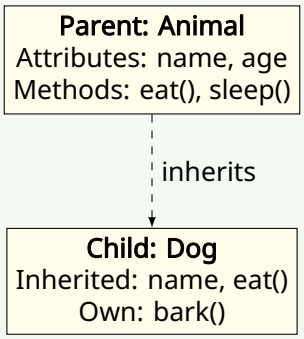
``ફાઇલ બનાવો, ફંક્શન્સ ડિફાઇન કરો, ઇમ્પોર્ટ કરો, વાપરો''

પ્રશ્ન 5(a) [3 ગુણ]

ઉદાહરણ સાથે Python માં સિંગલ ઇન્હેરિટન્સ સમજાવો.

જવાબ

Single inheritance એ જ્યારે એક class બરાબર એક parent class પાસેથી properties અને methods inherit કરે છે.
ઇન્હેરિટન્સ માળખું: Parent Class (Base) → Child Class (Derived).
ઇન્હેરિટન્સ આકૃતિ:



આકૃતિ 12. Single Inheritance

ઉદાહરણ કોડ:

```
class Animal:
    def eat(self):
        print("Eating")

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

my_dog = Dog()
my_dog.eat()   # Inherited
my_dog.bark()  # Own
```

મેમરી ટ્રીક

``એક Parent, એક Child''

પ્રશ્ન 5(b) [4 ગુણ]

Python માં એબ્સ્ટ્રેક્શનની વિભાવના અને તેના લાભો સમજાવો.

જવાબ

Abstraction જટિલ implementation details છુપાવે છે અને user ને માત્ર આવશ્યક features બતાવે છે.
એબ્સ્ટ્રેક્શન કન્સેપ્ટ્સ:

- **Abstract Class**: Instantiate કરી શકાતું નથી (class Shape(ABC):)
- **Abstract Method**: Implement કરવું જ પડે (@abstractmethod)

Implementation:

```
from abc import ABC, abstractmethod

class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

class Rectangle(Shape):
    def area(self):
        return self.length * self.width
```

લાભો:

- **સરળતા**: જટિલ details છુપાવે
- **સુરક્ષા**: આંતરિક implementation છુપાવે
- **જાળવણીયોગ્યતા**: Implementation બદલી શકાય

મેમરી ટ્રીક

``વિગતો છુપાવો, Interface બતાવો''

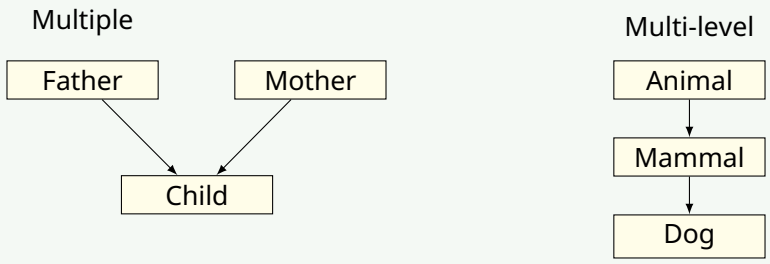
પ્રશ્ન 5(c) [7 ગુણ]

મલ્ટિપલ અને મલ્ટિ-લેવલ ઇન્હેરિટન્સનું કાર્ય દર્શાવતો Python પ્રોગ્રામ વિકસાવો.

જવાબ

પ્રોગ્રામ બંને inheritance types દર્શાવે છે: multiple (અનેક parents) અને multi-level (inheritance ની chain).

ઇન્હેરિટન્સ પદાનુક્રમ:



આકૃતિ 13. Inheritance Types

સંપૂર્ણ પ્રોગ્રામ:

```
print("=== Multi-level Inheritance ===")
class Animal:
    def eat(self): print("Eating")
class Mammal(Animal):
    def breathe(self): print("Breathing")
class Dog(Mammal):
    def bark(self): print("Barking")

d = Dog()
d.eat(); d.breathe(); d.bark()

print("\n=== Multiple Inheritance ===")
```

```python
class Father:
    def f_method(self): print("Father")
class Mother:
    def m_method(self): print("Mother")
class Child(Father, Mother):
    pass

c = Child()
c.f_method(); c.m_method()
```

મેમરી ટ્રીક

``અનેક Parents, મલ્ટિ-લેવલ Chain''

પ્રશ્ન 5(a OR) [3 ગુણ]

Python માં આવતી 3 પ્રકારની મેથડ્સનું કાર્ય સમજાવો.

જવાબ

Python classes માં ત્રણ પ્રકારની methods છે જે class data ને કેવી રીતે access કરે છે તેના આધારે.
મેથડ પ્રકારો ટેબલ:

કોષ્ટક 22. Method Types

મેથડ પ્રકાર	પ્રથમ Parameter	હેતુ
Instance Method	self	Instance data access
Class Method	cls	Class data access
Static Method	કોઈ નહીં	Utility functions

ઉદાહરણ કોડ:

```python
class Student:
    school = "ABC"
    def display(self): pass         # Instance
    @classmethod
    def get_school(cls): pass       # Class
    @staticmethod
    def is_adult(age): pass         # Static
```

મેમરી ટ્રીક

``Instance Self, Class Cls, Static કોઈ નહીં''

પ્રશ્ન 5(b OR) [4 ગુણ]

Python માં ઇન્હેરિટન્સ દ્વારા પોલીમોર્ફિઝમ સમજાવો.

જવાબ

Polymorphism વિવિધ classes ના objects ને સામાન્ય base class ના objects તરીકે treat કરવાની મંજૂરી આપે છે.
મુખ્ય કન્સેપ્ટ: સમાન method name, અલગ implementation.
ઉદાહરણ:

```
class Shape:
    def area(self): pass

class Rectangle(Shape):
    def area(self): return self.l * self.w

class Circle(Shape):
    def area(self): return 3.14 * self.r * self.r

shapes = [Rectangle(5,3), Circle(4)]
for s in shapes:
    print(s.area())  # Polymorphic call
```

- **લવચીકતા:** સમાન કોડ વિવિધ object types સાથે કામ કરે છે
- **વિસ્તરણશીલતા:** વર્તમાન કોડ બદલ્યા વિના નવા classes ઉમેરવા સરળ

મેમરી ટ્રીક

``સમાન નામ, અલગ વર્તન''

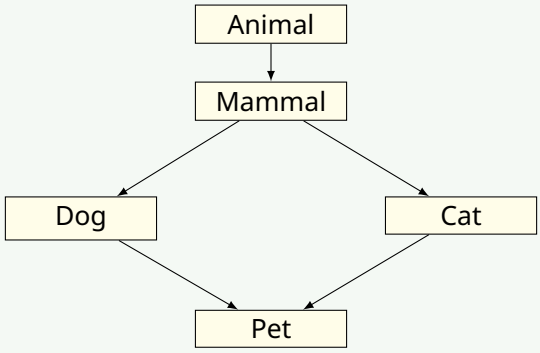
પ્રશ્ન 5(c OR) [7 ગુણ]

હાઇબ્રિડ ઇન્હેરિટન્સનું કાર્ય દર્શાવતો Python પ્રોગ્રામ વિકસાવો.

જવાબ

Hybrid inheritance એ single program structure માં multiple અને multi-level inheritance ને combine કરે છે.
માળખું:
- Animal → Mammal (Single)
- Mammal → Dog, Cat (Hierarchical)
- Dog, Cat → Pet (Multiple)

આકૃતિ:



આકૃતિ 14. Hybrid Inheritance

સંપૂર્ણ પ્રોગ્રામ:

```
# Hybrid Inheritance Demo
class Animal:
    def __init__(self, name): self.name = name

```

```python
class Mammal(Animal):
    def breathe(self): print("Breathing")

class Dog(Mammal):
    def bark(self): print("Barking")

class Cat(Mammal):
    def meow(self): print("Meowing")

class Pet(Dog, Cat):
    def play(self): print("Playing")

# Usage
p = Pet("Buddy")
p.breathe()  # From Mammal
p.bark()     # From Dog
p.meow()     # From Cat
p.play()     # Own
```