

Python Programming (4311601)

Milav Dabgar

Winter 2023

પ્રશ્ન પ્રશ્ન 1(અ) [03 ગુણ]

ફ્લો ચાર્ટ શું છે? ફ્લો ચાર્ટમાં વપરાતા પ્રતીકોની યાદી બનાવો.

જવાબ

ફ્લો ચાર્ટ એ અલ્ગોરિથમની ગ્રાફિકલ રજૂઆત છે જે પ્રક્રિયાના પગલાંઓ અને નિર્ણય બિંદુઓ દર્શાવે છે.

કોષ્ટક 1. ફ્લો ચાર્ટ પ્રતીકો

પ્રતીક	નામ	ઉપયોગ
અંડાકાર	ટર્મિનલ	પ્રારંભ/અંત
લંબચોરસ	પ્રોસેસ	પ્રક્રિયા/ગણતરી
હીરો	નિર્ણય	શરતી નિવેદનો
સમાંતર ચતુષ્કોણ	ઇનપુટ/આઉટપુટ	ડેટા લેવો/આપવો
વૃત્ત	કનેક્ટર	ભાગોને જોડવા
તીર	ફ્લો લાઇન	દિશા

મુખ્ય બિંદુઓ:

- વિઝ્યુઅલ રજૂઆત: પ્રોગ્રામ લોજિક ગ્રાફિકલી દર્શાવે
- પગલાં દર પગલાં: ક્રમિક ઓપરેશનનો ફ્લો
- નિર્ણય લેવો: હીરા શરતી શાખાઓ દર્શાવે

મેમરી ટ્રીક

“ફ્લો ચાર્ટ્સ પ્રોગ્રામ સ્ટેપ્સ વિઝ્યુઅલી દર્શાવે”

પ્રશ્ન પ્રશ્ન 1(બ) [04 ગુણ]

for લૂપ માટે ટૂંકી નોંધ લખો.

જવાબ

for લૂપ Python માં સિક્વન્સ (list, tuple, string, range) પર iterate કરવા માટે વપરાય છે.

કોષ્ટક 2. For લૂપ ટેબલ

ઘટક	સિન્ટેક્સ	ઉદાહરણ
મૂળભૂત	for variable in sequence:	for i in range(5):
રેન્જ	range(start, stop, step)	range(1, 10, 2)
યાદી	for item in list:	for x in [1,2,3]:
સ્ટ્રિંગ	for char in string:	for c in "hello":

સરળ કોડ ઉદાહરણ:

```
1 for i in range(3):
2     print(i)
3 # આઉટપુટ: 0, 1, 2
```

મુખ્ય લક્ષણો:

- ઓટોમેટિક iteration: મેન્યુઅલ કાઉન્ટરની જરૂર નથી
- સિકવન્સ ટ્રાવર્સલ: કોઈપણ iterable ઓબ્જેક્ટ સાથે કામ કરે
- રેન્જ ફંક્શન: નંબર સિકવન્સ સરળતાથી બનાવે

મેમરી ટ્રીક

“For લૂપ્સ સિકવન્સમાં iterate કરે”

પ્રશ્ન પ્રશ્ન 1(ક) [07 ગુણ]

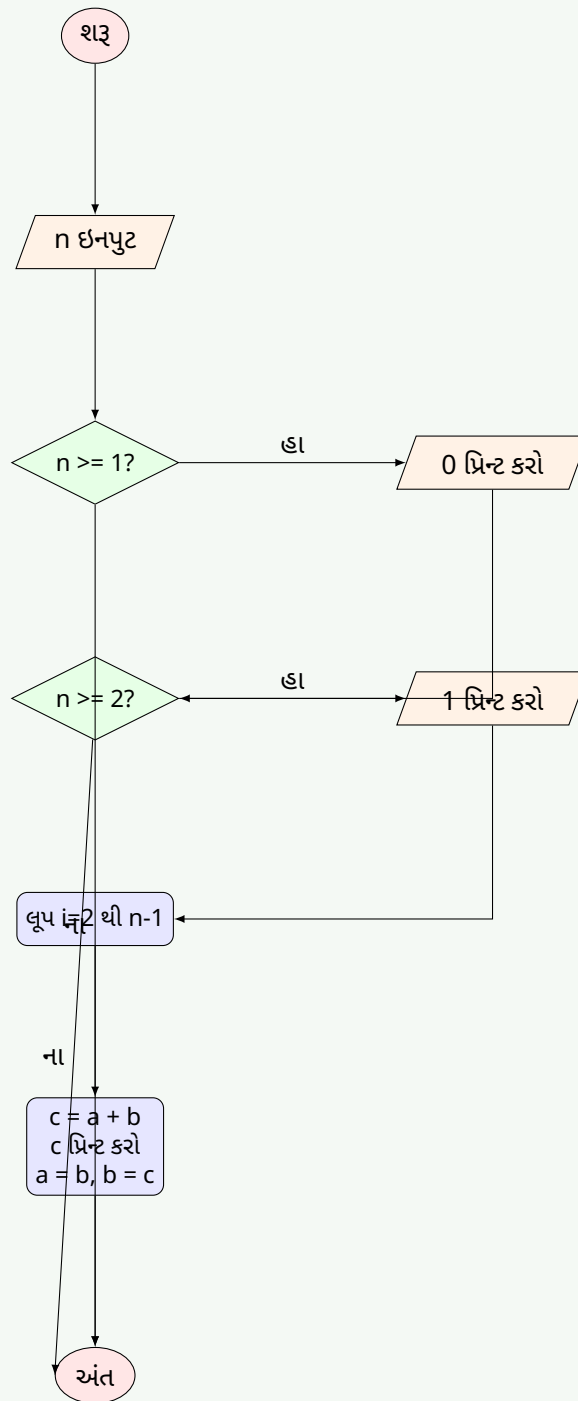
ફિબોનાકી શ્રેણીને nમી ટર્મ સુધી દર્શાવવા માટે એક પ્રોગ્રામ લખો જ્યાં યુઝર દ્વારા n આપવામાં આવે છે.

જવાબ

ફિબોનાકી શ્રેણી પ્રોગ્રામ:

```
1 # યુઝર પાસેથી ટર્મ્સની સંખ્યા લો
2 n = int(input("ટર્મ્સની સંખ્યા દાખલ કરો: "))
3
4 # પ્રથમ બે ટર્મ્સ initialize કરો
5 a, b = 0, 1
6
7 # પ્રથમ ટર્મ દર્શાવો
8 if n >= 1:
9     print(a, end=" ")
10
11 # બીજી ટર્મ દર્શાવો
12 if n >= 2:
13     print(b, end=" ")
14
15 # બાકીની ટર્મ્સ જનરેટ કરો
16 for i in range(2, n):
17     c = a + b
18     print(c, end=" ")
19     a, b = b, c
```

અલ્ગોરિથમ ફ્લો:



મુખ્ય કોન્સેપ્ટ્સ:

- સિક્વેન્શિયલ જનરેશન: દરેક ટર્મ = પાછલી બે ટર્મનો સરવાળો
- વેરિયેબલ સ્વેપિંગ: a, b વેલ્યુઝ અસરકારક રીતે અપડેટ કરો
- ચુઝર ઇનપુટ: ડાયનેમિક શ્રેણી લેન્થ

મેમરી ટ્રીક

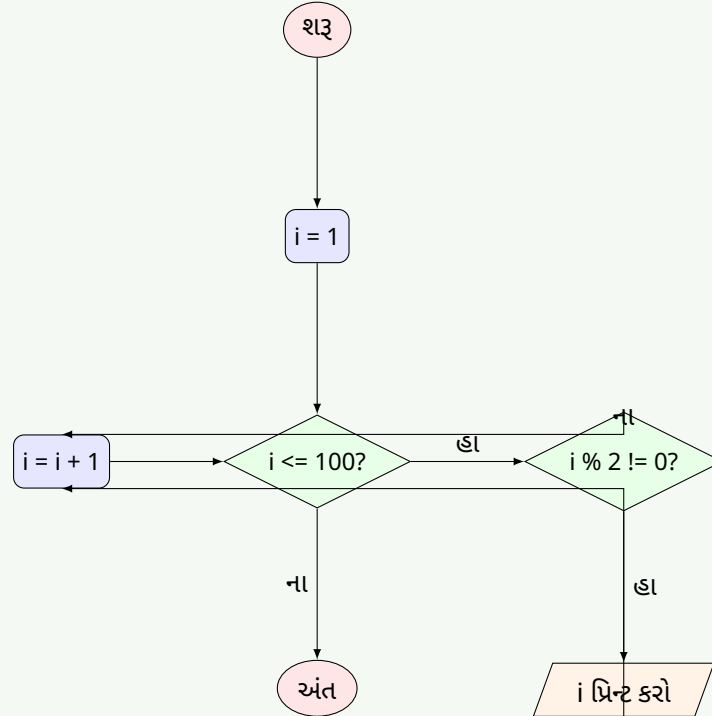
“ફિબોનાકી: પાછલા બે નંબરો ઉમેરો”

પ્રશ્ન પ્રશ્ન 1(ક OR) [07 ગુણ]

1 થી 100 સુધીના ODD નંબરો પ્રિન્ટ કરવા માટે ફ્લો ચાર્ટ દોરો.

જવાબ

1 થી 100 ODD નંબરો માટે ફ્લોચાર્ટ:



અનુસંગિક Python કોડ:

```

1 for i in range(1, 101):
2     if i % 2 != 0:
3         print(i, end=" ")
  
```

વૈકલ્પિક પદ્ધતિ:

```

1 for i in range(1, 101, 2):
2     print(i, end=" ")
  
```

મુખ્ય તત્વો:

- લૂપ કંટ્રોલ: i 1 થી 100 સુધી
- વિષમ ચેક: i % 2 != 0 શરત
- સ્ટેપ વધારો: આગલા નંબર પર જાઓ

મેમરી ટ્રીક

“વિષમ નંબરો: 2 થી ભાગ્યે 1 બાકી”

પ્રશ્ન પ્રશ્ન 2(અ) [03 ગુણ]

નંબર પેલિન્ડ્રોમ છે કે નહીં તે શોધવા માટે પ્રોગ્રામ લખો.

જવાબ

પેલિન્ડ્રોમ ચેક પ્રોગ્રામ:

```

1 # નંબર ઇનપુટ
2 num = int(input("નંબર દાખલ કરો: "))
3 temp = num
4 reverse = 0
5
6 # નંબરને રવિર્સ કરો
7 while temp > 0:
8     reverse = reverse * 10 + temp % 10
9     temp = temp // 10
10
11 # પેલિન્ડ્રોમ ચેક કરો
12 if num == reverse:
13     print(f"{num} પેલિન્ડ્રોમ છે")
14 else:
15     print(f"{num} પેલિન્ડ્રોમ નથી")

```

અલ્ગોરિથમ ટેબલ:

કોષ્ટક 3. અલ્ગોરિથમ

પગલું	ઓપરેશન	ઉદાહરણ (121)
1	છેલ્લો અંક મેળવો	$121 \% 10 = 1$
2	રિવર્સ બનાવો	$0 * 10 + 1 = 1$
3	છેલ્લો અંક દૂર કરો	$121 // 10 = 12$
4	0 સુધી પુનરાવર્તન	પ્રક્રિયા ચાલુ રાખો

મેમરી ટ્રીક

“પેલિન્ડ્રોમ આગળ પાછળ સરખું વાંચાય”

પ્રશ્ન પ્રશ્ન 2(બ) [04 ગુણ]

Python પ્રોગ્રામિંગની વિશેષતાઓ સમજાવો.

જવાબ

Python વિશેષતાઓનું ટેબલ:

કોષ્ટક 4. Python વિશેષતાઓ

વિશેષતા	વર્ણન	ફાયદો
સરળ સિન્ટેક્સ	સાદો, વાંચી શકાય તેવો કોડ	ઝડપી ડેવલપમેન્ટ
ઇન્ટરપ્રિટેડ	કમ્પાઇલેશનની જરૂર નથી	ઝડપી ટેસ્ટિંગ
ઓબ્જેક્ટ-ઓરિએન્ટેડ	ક્લાસ અને ઓબ્જેક્ટ સપોર્ટ	કોડ રિયુઝેબિલિટી
ઓપન સોર્સ	વાપરવા માટે ફ્રી	લાઇસન્સિંગ કોસ્ટ નથી
ક્રોસ-પ્લેટફોર્મ	મલ્ટિપલ OS પર ચાલે	વ્યાપક કમ્પેટિબિલિટી
મોટી લાઇબ્રેરીઓ	વ્યાપક બિલ્ટ-ઇન મોડ્યુલ્સ	સમૃદ્ધ કાર્યક્ષમતા

મુખ્ય ફાયદાઓ:

- શિખાઉ-મિત્ર: શીખવામાં અને સમજવામાં સરળ
- વર્સેટાઇલ: વેબ ડેવલપમેન્ટ, AI, ડેટા સાયન્સ
- કોમ્યુનિટી સપોર્ટ: મોટો ડેવલપર કોમ્યુનિટી

- સાયનેમિક ટાઇપિંગ: વેરિયેબલ ટાઇપ ડિસ્કવેરેશનની જરૂર નથી

મેમરી ટ્રીક

"Python: સરળ, શક્તિશાળી, લોકપ્રિય પ્રોગ્રામિંગ"

પ્રશ્ન પ્રશ્ન 2(ક) [07 ગુણ]

Python પ્રોગ્રામની બેસિક સ્ટ્રક્ચર સમજાવો.

જવાબ

Python પ્રોગ્રામ સ્ટ્રક્ચર:

```

1  #!/usr/bin/env python3
2  # Shebang લાઇન વૈકલ્પિક()
3
4  """ડોક્યુમેન્ટેશન
5  સ્ટ્રિંગ (docstring) પ્રોગ્રામનો
6  હેતુ વર્ણવે છે
7  """
8
9  # Import સ્ટેટમેન્ટ્સ
10 import math
11 from datetime import date
12
13 # ગ્લોબલ વેરિયેબલ્સ
14 PI = 3.14159
15 count = 0
16
17 # ફંક્શન ડેફિનીશિન્સ
18 def calculate_area(radius):
19     """વર્તુળનો ક્ષેત્રફળ કેલ્ક્યુલેટ કરે"""
20     return PI * radius * radius
21
22 # ક્લાસ ડેફિનીશિન્સ
23 class Calculator:
24     def __init__(self):
25         self.result = 0
26
27 # મેઇન પ્રોગ્રામ એક્ઝિક્યુશન
28 if __name__ == "__main__":
29     # પ્રોગ્રામ લોજિક અહીં
30     radius = 5
31     area = calculate_area(radius)
32     print(f"ક્ષેત્રફળ: {area}")

```

સ્ટ્રક્ચર કમ્પોનન્ટ્સ ટેબલ:

કોષ્ટક 5. સ્ટ્રક્ચર

ઘટક	હેતુ	ઉદાહરણ
Shebang	સિસ્ટમ ઇન્ટરપ્રિટર	#!/usr/bin/env python3
Docstring	પ્રોગ્રામ દસ્તાવેજીકરણ	"""Program description"""
Imports	બાહ્ય મોડ્યુલ્સ	import math
Variables	ગ્લોબલ ડેટા સ્ટોરેજ	PI = 3.14159
Functions	પુનઃઉપયોગી કોડ બ્લોક્સ	def function_name():

મેમરી ટ્રીક

`સ્ટ્રિંગ્સ: ઇમ્પોર્ટ, ડિક્ઝાઇન, એક્ટિવિટી`

પ્રશ્ન પ્રશ્ન 2(બ OR) [03 ગુણ]

સ્ટ્રિંગ્સને રિવર્સ કરવા માટે પ્રોગ્રામ લખો.

જવાબ

સ્ટ્રિંગ રિવર્સલ પ્રોગ્રામ:

```

1 # પદ્ધતિ 1: સ્લાઇસિંગ વાપરીને
2 string = input("સ્ટ્રિંગ દાખલ કરો: ")
3 reversed_string = string[::-1]
4 print(f"રિવર્સ: {reversed_string}")
5
6 # પદ્ધતિ 2: લૂપ વાપરીને
7 string = input("સ્ટ્રિંગ દાખલ કરો: ")
8 reversed_string = ""
9 for char in string:
10     reversed_string = char + reversed_string
11 print(f"રિવર્સ: {reversed_string}")

```

રિવર્સલ પદ્ધતિઓનું ટેબલ:

કોષ્ટક 6. રિવર્સલ પદ્ધતિઓ

પદ્ધતિ	સિન્ટેક્સ	ઉદાહરણ
સ્લાઇસિંગ	string[::-1]	"hello" → "olleh"
લૂપ	કેરેક્ટર દ્વારા કેરેક્ટર બનાવો	દરેક char આગળ ઉમેરો
બિલ્ટ-ઇન	"".join(reversed(string))	રિવર્સ સિક્વન્સ જોડો

મેમરી ટ્રીક

`રિવર્સ: છેલ્લો કેરેક્ટર પહેલો`

પ્રશ્ન પ્રશ્ન 2(બ OR) [04 ગુણ]

લોજિકલ ઓપરેટર્સને ઉદાહરણ સાથે સમજાવો.

જવાબ

Python લોજિકલ ઓપરેટર્સ:

કોષ્ટક 7. લોજિકલ ઓપરેટર્સ

ઓપરેટર	સિમ્બોલ	વર્ણન	ઉદાહરણ	પરિણામ
AND	and	બંને શરતો સાચી	True and False	False
OR	or	ઓછામાં ઓછી એક શરત સાચી	True or False	True
NOT	not	શરતની વિરુદ્ધ	not True	False

ઉદાહરણ કોડ:

```

1 a = 10
2 b = 5
3
4 # AND ઓપરેટર
5 if a > 5 and b < 10:
6     print("બંને શરતો સાચી")
7
8 # OR ઓપરેટર
9 if a > 15 or b < 10:
10    print("ઓછામાં ઓછી એક શરત સાચી")
11
12 # NOT ઓપરેટર
13 if not (a < 5):
14    print("a 5 કરતાં નાનું નથી")

```

ટ્રુથ ટેબલ:

કોષ્ટક 8. ટ્રુથ ટેબલ

A	B	A and B	A or B	not A
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

મેમરી ટ્રીક

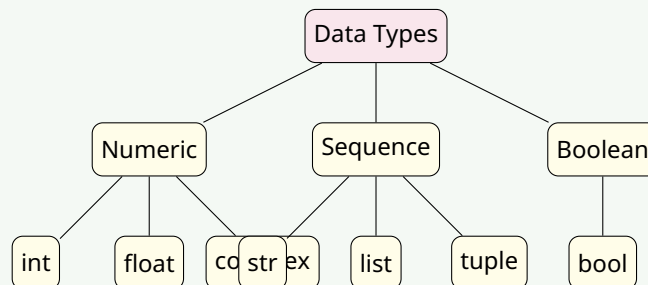
“AND બધાની જરૂર, OR એકની જરૂર, NOT ઉલટાવે”

પ્રશ્ન પ્રશ્ન 2(ક OR) [07 ગુણ]

Python માં વિવિધ ડેટા પ્રકારો સમજાવો

જવાબ

Python ડેટા ટાઇપ્સ વર્ગીકરણ:



ડેટા ટાઇપ્સ ટેબલ:

કોષ્ટક 9. ડેટા ટાઇપ્સ

ટાઇપ	ઉદાહરણ	વર્ણન	Mutable
int	42	પૂર્ણ સંખ્યાઓ અને ના	
float	3.14	દશાંશ સંખ્યાઓ	ના
str	"hello"	ટેક્સ્ટ ડેટા	ના
list	[1,2,3]	ક્રમાંકિત સંગ્રહ	હા
tuple	(1,2,3)	ક્રમાંકિત અપરિવર્તનીય	ના
dict	{"a":1}	કી-વેલ્યુ જોડીઓ	હા
bool	True/False	બુલિયન વેલ્યુઝ	ના
set	{1,2,3},	યુનિક તત્વો	હા

મેમરી ટ્રીક

“Python ટાઇપ્સ: નંબર્સ, સિક્વન્સીસ, કલેક્શન્સ”

પ્રશ્ન પ્રશ્ન 3(અ) [03 ગુણ]

Python માં ફ્લો કંટ્રોલ શું છે? ઉદાહરણ સાથે સમજાવો

જવાબ

ફ્લો કંટ્રોલ શરતી અને લૂપ સ્ટ્રક્ચર્સ વાપરીને પ્રોગ્રામ સ્ટેટમેન્ટ્સનો એક્ઝિક્યુશન ઓર્ડર મેનેજ કરે છે.
ફ્લો કંટ્રોલ પ્રકારોનું ટેબલ:

કોષ્ટક 10. ફ્લો કંટ્રોલ

પ્રકાર	સ્ટેટમેન્ટ	હેતુ	ઉદાહરણ
સિક્વેન્શિયલ	સામાન્ય એક્ઝિક્યુશન	લાઇન બાય લાઇન	print("Hello")
સિલેક્શન	if, elif, else	નિર્ણય લેવો	if x > 0:
Iteration	for, while	પુનરાવર્તન	for i in range(5):
Jump	break, continue	લૂપ કંટ્રોલ	break

મેમરી ટ્રીક

“ફ્લો કંટ્રોલ: નિર્ણય, પુનરાવર્તન, Jump”

પ્રશ્ન પ્રશ્ન 3(બ) [04 ગુણ]

નેસ્ટેડ if સ્ટેટમેન્ટ સમજાવવા માટે પ્રોગ્રામ લખો.

જવાબ

નેસ્ટેડ If સ્ટેટમેન્ટ પ્રોગ્રામ:

```

1 # નેસ્ટેડ if વાપરીને ગ્રેડ કેલ્ક્યુલેશન
2 marks = int(input("માર્ક્સ દાખલ કરો: "))
3
4 if marks >= 0 and marks <= 100:
5     if marks >= 90:
6         grade = "A+"

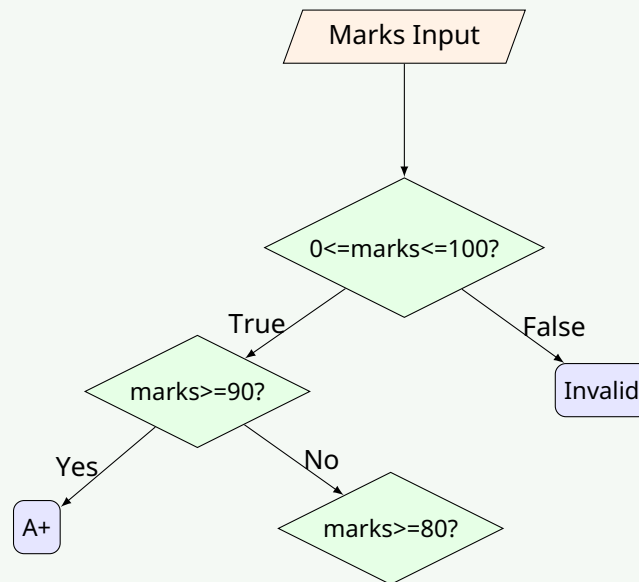
```

```

7 elif marks >= 80:
8     if marks >= 85:
9         grade = "A"
10    else:
11        grade = "B+"
12 elif marks >= 70:
13     grade = "B"
14 elif marks >= 60:
15     grade = "C"
16 else:
17     grade = "F"
18 print(f"ગ્રેડ: {grade}")
19 else:
20     print("અયોગ્ય માર્ક્સ")

```

નેસ્ટેડ સ્ટ્રક્ચર ડાયાગ્રામ:



મેમરી ટ્રીક

“નેસ્ટેડ If: નિર્ણયોની અંદર નિર્ણયો”

પ્રશ્ન પ્રશ્ન 3(ક) [07 ગુણ]

Arguments અને Parameters ના પ્રકારો સમજાવવા માટે એક પ્રોગ્રામ લખો.

જવાબ

Arguments અને Parameters ના પ્રકારો:

```

1 # 1. પોઝિશનલ Arguments
2 def greet(name, age):
3     print(f"હેલો {name}, તમારી ઉંમર {age} વર્ષ છે")
4
5 greet("જોન", 25) # પોઝિશનલ arguments
6
7 # 2. કીવર્ડ Arguments
8 greet(age=30, name="એલિસ") # કીવર્ડ arguments
9

```

```

10 # 3. ડફિનિટ Parameters
11 def introduce(name, city="અજાણ"):
12     print(f"{name} {city} માં રહે છે")
13
14 introduce("બોબ") # ડફિનિટ વેલ્યુ વાપરે
15 introduce("કેરોલ", "મુંબઈ") # ડફિનિટ ઓવરરાઇડ
16
17 # 4. વેરિયેબલ નંથ- Arguments (*args)
18 def sum_all(*numbers):
19     return sum(numbers)
20
21 result = sum_all(1, 2, 3, 4, 5)
22 print(f"સરવાળો: {result}")
23
24 # 5. કીવર્ડ વેરિયેબલ Arguments (**kwargs)
25 def display_info(**info):
26     for key, value in info.items():
27         print(f"{key}: {value}")
28
29 display_info(name="ડેવિડ", age=28, city="બોસ્ટન")

```

મેમરી ટ્રીક

"Parameters: પોઝિશન, કીવર્ડ્સ, ડિફોલ્ટ્સ, વેરિયેબલ્સ"

પ્રશ્ન પ્રશ્ન 3(અ OR) [03 ગુણ]

break અને continue statement ને ઉદાહરણ સાથે સમજાવો.

જવાબ

Break અને Continue સ્ટેટમેન્ટ્સ:

Break સ્ટેટમેન્ટ:

```

1 # Break ઉદાહરણ - લૂપમાંથી બહાર નીકળો
2 for i in range(10):
3     if i == 5:
4         break
5     print(i)
6 # આઉટપુટ: 0, 1, 2, 3, 4

```

Continue સ્ટેટમેન્ટ:

```

1 # Continue ઉદાહરણ - iteration છોડો
2 for i in range(5):
3     if i == 2:
4         continue
5     print(i)
6 # આઉટપુટ: 0, 1, 3, 4

```

સરખામણી ટેબલ:

કોષ્ટક 11. Break vs Continue

સ્ટેટમેન્ટ	હેતુ	ક્રિયા	ઉદાહરણ ઉપયોગ
break	લૂપમાંથી બહાર નીકળો	સંપૂર્ણ લૂપ સમાપ્ત કરે	શરત પર બહાર નીકળો
continue	iteration છોડો	આગલા iteration પર જાઓ	સ્પેસિફિક વેલ્યુઝ છોડો

મેમરી ટ્રીક

“Break બહાર નીકળે, Continue છોડે”

પ્રશ્ન પ્રશ્ન 3(બ OR) [04 ગુણ]

નીચેની પેટર્ન દર્શાવવા માટે એક પ્રોગ્રામ બનાવો

જવાબ

પેટર્ન:

```
1
12
123
1234
12345
```

નંબર પેટર્ન પ્રોગ્રામ:

```
1 # પદ્ધતિ 1: નેસ્ટેડ લૂપ્સ વાપરીને
2 rows = 5
3 for i in range(1, rows + 1):
4     for j in range(1, i + 1):
5         print(j, end="")
6     print() # નવી લાઇન
7
8 # પદ્ધતિ 2: સ્ટ્રિંગ મેનિપ્યુલેશન વાપરીને
9 for i in range(1, 6):
10     line = ""
11     for j in range(1, i + 1):
12         line += str(j)
13     print(line)
```

મેમરી ટ્રીક

“પેટર્ન: પંક્તિ નંબર કોલમ કાઉન્ટ નક્કી કરે”

પ્રશ્ન પ્રશ્ન 3(ક OR) [07 ગુણ]

દરેક માટે કોડ લખીને નીચેના ગાણિતિક કાર્યો સમજાવો: 1. abs() 2. max() 3. pow() 4. sum()

જવાબ

Python માં ગાણિતિક ફંક્શન્સ:

```
1 # 1. abs() - એબ્સોલ્યુટ વેલ્યુ
2 numbers = [-5, 3.7, -10.2, 0]
3 print("abs() ફંક્શન ઉદાહરણો:")
4 for num in numbers:
5     print(f"abs({num}) = {abs(num)}")
6
7 # 2. max() - મહત્તમ વેલ્યુ
```

```

8 list1 = [4, 7, 2, 9, 1]
9 print(f"\nmax() ફંક્શન ઉદાહરણો:")
10 print(f"max({list1}) = {max(list1)}")
11 print(f"max(10, 25, 5) = {max(10, 25, 5)}")
12
13 # 3. pow() - પાવર ફંક્શન
14 print(f"\npow() ફંક્શન ઉદાહરણો:")
15 print(f"pow(2, 3) = {pow(2, 3)}") # 2^3 = 8
16 print(f"pow(5, 2) = {pow(5, 2)}") # 5^2 = 25

```

મેમરી ટ્રીક

"Math ફંક્શન્સ: એબ્સોલ્યુટ, મહત્તમ, પાવર, સરવાળો"

પ્રશ્ન પ્રશ્ન 4(અ) [03 ગુણ]

Variables નો scope સમજાવો.

જવાબ

વેરિયેબલ સ્કોપ એ પ્રોગ્રામમાં તે પ્રદેશનો સંદર્ભ આપે છે જ્યાં વેરિયેબલ એક્સેસ કરી શકાય. સ્કોપ પ્રકારોનું ટેબલ:

કોષ્ટક 12. સ્કોપ પ્રકારો

સ્કોપ	વર્ણન	જીવનકાળ	એક્સેસ
લોકલ	ફંક્શનની અંદર	ફંક્શન એક્ઝિક્યુશન	માત્ર ફંક્શન
ગ્લોબલ	ફંક્શનોની બહાર	પ્રોગ્રામ એક્ઝિક્યુશન	આખો પ્રોગ્રામ
બિલ્ટ-ઇન	Python કીવર્ડ્સ	Python સેશન	બધે

મેમરી ટ્રીક

"સ્કોપ: લોકલ ફંક્શનમાં રહે, ગ્લોબલ બધે રહે"

પ્રશ્ન પ્રશ્ન 4(બ) [04 ગુણ]

નેસ્ટેડ LOOP બનાવવા અને નંબરો દર્શાવવા માટે એક પ્રોગ્રામ વિકસાવો.

જવાબ

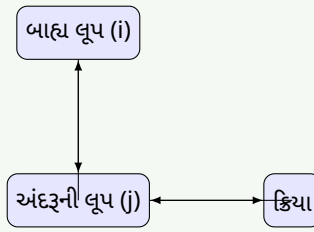
નેસ્ટેડ લૂપ પ્રોગ્રામ:

```

1 # ઉદાહરણ 1: નંબર ગ્રીડ
2 print("નંબર ગ્રીડ પેટર્ન:")
3 for i in range(1, 4):
4     for j in range(1, 5):
5         print(f"{i}-{j}", end=" ")
6     print() # દરેક પંક્તિપછી નવી લાઇન

```

નેસ્ટેડ લૂપ સ્ટ્રક્ચર:



મેમરી ટ્રીક

“નેસ્ટેડ લૂપ્સ: બાહ્ય નિયંત્રણ કરે અંદરની”

પ્રશ્ન પ્રશ્ન 4(ક) [07 ગુણ]

1 થી 50 ની રેન્જમાં ODD અને EVEN નંબરોની યાદી બનાવવા માટે પ્રોગ્રામ લખો.

જવાબ

ODD અને EVEN નંબરો પ્રોગ્રામ:

```

1 # પદ્ધતિ 1: લૂપ્સ અને શરતો વાપરીને
2 odd_numbers = []
3 even_numbers = []
4
5 for i in range(1, 51):
6     if i % 2 == 0:
7         even_numbers.append(i)
8     else:
9         odd_numbers.append(i)
10
11 print("ODD નંબરો (1-50):")
12 print(odd_numbers)
13
14 print("\nEVEN નંબરો (1-50):")
15 print(even_numbers)
  
```

મેમરી ટ્રીક

“Odd/Even: 2 થી ભાગતા બાકી 1/0”

પ્રશ્ન પ્રશ્ન 4(અ OR) [03 ગુણ]

ઉદાહરણ સાથે String Slicing સમજાવો.

જવાબ

સ્ટ્રિંગ સ્લાઇસિંગ [start:stop:step] સિન્ટેક્સ વાપરીને સ્ટ્રિંગના ભાગો કાઢે છે.
સ્લાઇસિંગ સિન્ટેક્સ ટેબલ:

કોષ્ટક 13. સ્લાઇસિંગ

સિન્ટેક્સ	વર્ણન	ઉદાહરણ	પરિણામ
s[start:stop]	શરૂઆતથી stop-1 સુધી	"hello"[1:4]	"ell"
s[start:]	શરૂઆતથી અંત સુધી	"hello"[2:]	"llo"
s[:stop]	શરૂઆતથી stop-1 સુધી	"hello"[:3]	"hel"
s[::step]	દરેક step કેરેક્ટર	"hello"[:2]	"hlo"
s[::-1]	રિવર્સ સ્ટ્રિંગ	"hello"[::-1]	"olleh"

મેમરી ટ્રીક

``સ્લાઇસ: સ્ટાર્ટ, સ્ટોપ, સ્ટેપ``

પ્રશ્ન પ્રશ્ન 4(બ OR) [04 ગુણ]

આપેલ નંબરનું ફેક્ટોરિયલ શોધવા માટે યુઝર ડિફાઇન્ડ ફંક્શન વાપરીને પ્રોગ્રામ લખો.

જવાબ**ફેક્ટોરિયલ ફંક્શન પ્રોગ્રામ:**

```

1 def factorial(n):
2     """રકિર્જન વાપરીને ફેક્ટોરિયલ"""
3     if n == 0 or n == 1:
4         return 1
5     else:
6         return n * factorial(n - 1)
7
8 # મેઇન પ્રોગ્રામ
9 number = int(input("નંબર દાખલ કરો: "))
10 if number < 0:
11     print("નેગેટિવ નંબરો માટે ફેક્ટોરિયલ નથી")
12 else:
13     result1 = factorial(number)
14     print(f"{number} નું ફેક્ટોરિયલ = {result1}")

```

મેમરી ટ્રીક

``ફેક્ટોરિયલ: નીચેના તમામ નંબરો ગુણો``

પ્રશ્ન પ્રશ્ન 4(ક OR) [07 ગુણ]

આપેલ સ્ટ્રિંગમાં સબ સ્ટ્રિંગ હાજર છે કે નહીં તે તપાસવા માટે યુઝર ડિફાઇન્ડ ફંક્શન લખો.

જવાબ**સબસ્ટ્રિંગ ચેક ફંક્શન:**

```

1 def find_substring(main_string, sub_string):
2     """મેઇન સ્ટ્રિંગમાં સબસ્ટ્રિંગ છે કે નહીં તે તપાસો"""
3     if sub_string in main_string:
4         index = main_string.find(sub_string)
5         return True, index

```

```

6     else:
7         return False, -1
8
9     # મેઇન પ્રોગ્રામ
10    text = input("મેઇન સ્ટ્રિંગ દાખલ કરો: ")
11    search = input("સર્ચ કરવા માટે સબસ્ટ્રિંગ: ")
12
13    found, position = find_substring(text, search)
14    if found:
15        print(f"સબસ્ટ્રિંગ '{search}' પોઝિશન {position} પર મળ્યું")
16    else:
17        print(f"સબસ્ટ્રિંગ '{search}' મળ્યું નથી")

```

મેમરી ટ્રીક

“સબસ્ટ્રિંગ: સર્ચ, ફાઇન્ડ, કાઉન્ટ, પોઝિશન”

પ્રશ્ન પ્રશ્ન 5(અ) [03 ગુણ]

List કેવી રીતે બનાવવું અને એક્સેસ કરવું તે ઉદાહરણ સાથે સમજાવો.

જવાબ

લિસ્ટ બનાવવું અને એક્સેસ કરવું:

```

1 # લિસ્ટ બનાવવું
2 numbers = [1, 2, 3, 4, 5]
3
4 # એક્સેસિંગ
5 print(f"પ્રથમ તત્વ: {numbers[0]}") # 1
6 print(f"છેલ્લું તત્વ: {numbers[-1]}") # 5
7 print(f"સ્લાઇસ: {numbers[1:4]}") # [2, 3, 4]

```

મેમરી ટ્રીક

“લિસ્ટ: ક્રિએટ, ઇન્ડેક્સ, એક્સેસ”

પ્રશ્ન પ્રશ્ન 5(બ) [04 ગુણ]

LIST પર કરી શકાતા ઓપરેશન્સની યાદી બનાવો. એક List બનાવી બીજા List માં કોપી કરવા માટે પ્રોગ્રામ લખો.

જવાબ

લિસ્ટ ઓપરેશન્સ અને કોપી પ્રોગ્રામ:

```

1 # ઓરિજિનલ લિસ્ટ
2 original = [1, 2, 3, 4, 5]
3 print(f"ઓરિજિનલ લિસ્ટ: {original}")
4
5 # કોપી પદ્ધતિઓ
6 shallow_copy = original.copy()
7 slice_copy = original[:]

```



```

8 list_copy = list(original)
9
10 # ઓરજિનલ બદલો
11 original.append(6)
12 print(f"એપેન્ડ પછી: {original}")
13 print(f"શેલો કોપી: {shallow_copy}")

```

લિસ્ટ ઓપરેશન્સ ટેબલ:

કોષ્ટક 14. લિસ્ટ ઓપરેશન્સ

ઓપરેશન	પદ્ધતિ	ઉદાહરણ	પરિણામ
Add	append()	[1,2].append(3)	[1,2,3]
Insert	insert()	[1,3].insert(1,2)	[1,2,3]
Remove	remove()	[1,2,3].remove(2)	[1,3]
Pop	pop()	[1,2,3].pop()	[1,2]

મેમરી ટ્રીક

“લિસ્ટ ઓપરેશન્સ: એડ, ઇન્સર્ટ, રિમૂવ, પોપ, કોપી”

પ્રશ્ન પ્રશ્ન 5(ક) [07 ગુણ]

LIST ના વિવિધ Built in methods ની યાદી અને ઉપયોગ આપો

જવાબ

બિલ્ટ-ઇન લિસ્ટ મેથડ્સ:

```

1 # ડેમો લસ્ટ
2 fruits = ['apple', 'banana', 'cherry', 'apple']
3
4 # મોડિફિકેશન
5 fruits.append('date')      # અંતે ઉમેરો
6 fruits.insert(1, 'avocado') # ઇન્ડેક્સ પર
7 fruits.remove('apple')    # પ્રથમ રમૂવ
8
9 # સર્ચ
10 count = fruits.count('apple') # ગણતરી
11 index = fruits.index('banana') # ઇન્ડેક્સ શોધો
12
13 # સોર્ટિંગ
14 fruits.sort()             # સોર્ટ
15 fruits.reverse()          # રવિર્સ

```

મેમરી ટ્રીક

“લિસ્ટ મેથડ્સ: એડ, રિમૂવ, સર્ચ, સોર્ટ, કોપી”

પ્રશ્ન પ્રશ્ન 5(અ OR) [03 ગુણ]

સ્ટ્રિંગ કેવી રીતે બનાવવી અને ટ્રાવર્સ કરવી તે ઉદાહરણ આપી સમજાવો.

જવાબ

સ્ટ્રિંગ ક્રિએશન અને ટ્રાવર્સલ:

```

1 # સ્ટ્રિંગ બનાવવી
2 string1 = "Hello World"    # ડબલ ક્વોટ્સ
3 string2 = 'Python'        # સિંગલ ક્વોટ્સ
4
5 # સ્ટ્રિંગ ટ્રાવર્સલ
6 text = "Python"
7
8 # પદ્ધતિ 1: લૂપ વાપરીને
9 for char in text:
10     print(char, end=" ")

```

મેમરી ટ્રીક

“સ્ટ્રિંગ્સ: કિએટ, લૂપ, એક્સેસ”

પ્રશ્ન પ્રશ્ન 5(બ OR) [04 ગુણ]

String પર કરી શકાતા ઓપરેશન્સની યાદી બનાવો. કોઈ પણ 2 ઓપરેશન્સ માટે કોડ લખો

જવાબ

સ્ટ્રિંગ ઓપરેશન્સ:

```

1 # ઓપરેશન 1: સ્ટ્રિંગ કન્કેટેનેશન
2 first = "જય"
3 last = "હનિદ"
4 full = first + " " + last
5 print(f"કન્કેટેનેશન: {full}")
6
7 # ઓપરેશન 2: કેસ કન્વર્ઝન
8 sentence = "learn python"
9 title_case = sentence.title()
10 print(f"ટાઇટલ કેસ: {title_case}")

```

મેમરી ટ્રીક

“સ્ટ્રિંગ ઓપરેશન્સ: જોઈન, કેસ, સ્પ્લિટ, ફાઇન્ડ”

પ્રશ્ન પ્રશ્ન 5(ક OR) [07 ગુણ]

String ના વિવિધ built – in methods ની યાદી અને ઉપયોગ આપો.

જવાબ

બિલ્ટ-ઇન સ્ટ્રિંગ મેથડ્સ:

```

1 # ડેમો સ્ટ્રિંગ
2 text = " Python Programming "
3

```

```
4 # કેસ
5 print(f"upper(): {text.upper()}")
6 print(f"lower(): {text.lower()}")
7
8 # વ્હાઇટસ્પેસ
9 print(f"strip(): '{text.strip()}'")
10
11 # સર્ચ
12 print(f"find('Python'): {text.find('Python')}")
13
14 # સ્પ્લિટ
15 words = text.split()
16 print(f"split(): {words}")
```

મેમરી ટ્રીક

“સ્ટ્રિંગ મેથડ્સ: કેસ, કલીન, ચેક, ચેન્જ”