Python Programming (4311601) - Winter 2024 Solution

Milav Dabgar

January 13, 2025

પ્રશ્ન પ્રશ્ન 1(અ) [03 ગુણ]

પ્રોબ્લમ સોલવિંગ, અલ્ગોરિધમ અને સ્યુડો કોડ વ્યાખ્યાયિત કરો.

જવાબ

વ્યાખ્યાઓ:
મુખ્ય મુદ્દાઓ:

- **પ્રોબ્લમ સોલવિંગ**: જટિલ સમસ્યાઓને વ્યવસ્થિત પગલાઓમાં વહેંચવું
- **અલ્ગોરિધમ**: મર્યાદિત, નિશ્ચિત, અસરકારક અને યોગ્ય આઉટપુટ આપતું હોવું જોઈએ
- **સ્યુડો કોડ**: માનવ ભાષા અને programming કોડ વચ્ચેનો સેતુ

મેમરી ટ્રીક

``PAP - Problem, Algorithm, Pseudo''

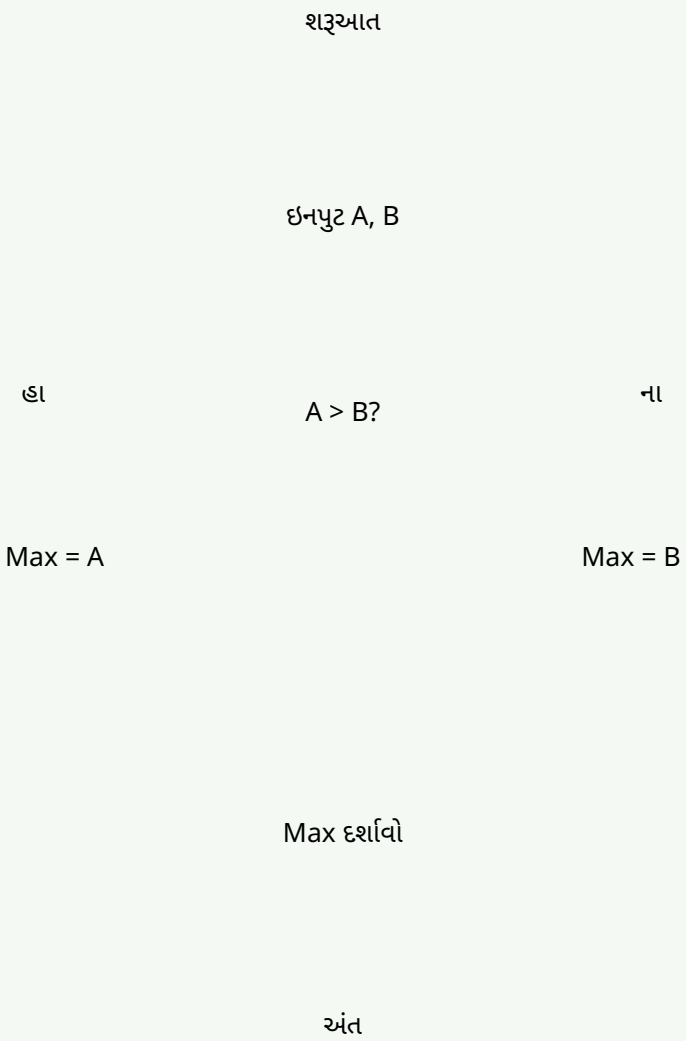
પ્રશ્ન પ્રશ્ન 1(બ) [04 ગુણ]

ફ્લોચાર્ટના જુદા જુદા સિમ્બોલ સમજાવો. બે નંબર માંથી મહત્તમ નંબર શોધતો ફ્લોચાર્ટ ડિઝાઇન કરો.

જવાબ

ફ્લોચાર્ટ સિમ્બોલ:

બે નંબરના મહત્તમ માટે ફ્લોચાર્ટ:



સમજૂતી:
- **શરૂઆત/અંત**: પ્રવેશ અને બહાર નીકળવાના બિંદુઓ
- **ઇનપુટ/આઉટપુટ**: ડેટા ફ્લો ઓપરેશન્સ
- **નિર્ણય**: શરતી branching
- **પ્રક્રિયા**: ગણતરીના પગલાં

મેમરી ટ્રીક

``SIPO - Start, Input, Process, Output''

પ્રશ્ન પ્રશ્ન 1(ક) [07 ગુણ]

પાયથોનના વિવિધ એરિથમેટિક ઓપરેટરોની યાદી બનાવો. વિવિધ એરિથમેટિક ઓપરેશન્સ માટેનો Python કોડ લખો.

જવાબ

એરિથમેટિક ઓપરેટરો:

કોડ:

```
a = 10
b = 3
print(f"Addition: {a + b}")
print(f"Subtraction: {a - b}")
print(f"Multiplication: {a * b}")
print(f"Division: {a / b}")
print(f"Floor Division: {a // b}")
print(f"Modulus: {a % b}")
print(f"Power: {a ** b}")
```

મેમરી ટ્રીક

``Add-Sub-Mul-Div-Floor-Mod-Pow''

પ્રશ્ન પ્રશ્ન 1(ક OR) [07 ગુણ]

પાયથોનના વિવિધ કંપેરિઝન ઓપરેટરોની યાદી બનાવો. વિવિધ કંપેરિઝન ઓપરેશન્સ માટેનો Python કોડ લખો.

જવાબ

કંપેરિઝન ઓપરેટરો:
કોડ:

```
x = 8
y = 5
print(f"Equal: {x == y}")
print(f"Not Equal: {x != y}")
print(f"Greater: {x > y}")
print(f"Less: {x < y}")
print(f"Greater Equal: {x >= y}")
print(f"Less Equal: {x <= y}")
```

મેમરી ટ્રીક

``Equal-Not-Greater-Less-GreaterEqual-LessEqual''

પ્રશ્ન પ્રશ્ન 2(અ) [03 ગુણ]

મેમ્બરશિપ ઓપરેટર્સ ઉપર ટૂંક નોંધ લખો.

જવાબ

મેમ્બરશિપ ઓપરેટર્સ:

મુખ્ય મુદ્દાઓ:
- **in ઓપરેટર**: જો એલિમેન્ટ sequence માં મળે તો True આપે
- **not in ઓપરેટર**: જો એલિમેન્ટ sequence માં ન મળે તો True આપે
- **ઉપયોગ**: Lists, strings, tuples, dictionaries માં

મેમરી ટ્રીક

``In-Not-In for membership testing''

પ્રશ્ન પ્રશ્ન 2(બ) [04 ગુણ]

પાયથોન વ્યાખ્યાયિત કરો. પાયથોન પ્રોગ્રામિંગની વિવિધ એપ્લિકેશનો લખો.

જવાબ

પાયથોન વ્યાખ્યા: સરળતા અને વાંચનીયતા માટે જાણીતી high-level, interpreted programming language.
એપ્લિકેશનો:
વિશેષતાઓ:
- **Interpreted**: compilation ની જરૂર નથી
- **Cross-platform**: બહુવિધ OS પર ચાલે છે
- **વિશાળ libraries**: વ્યાપક standard library

મેમરી ટ્રીક

``Web-Data-AI-Desktop-Games''

પ્રશ્ન પ્રશ્ન 2(ક) [07 ગુણ]

પાયથોન પ્રોગ્રામ લખો જે નીચેની વિગતોનો ઉપયોગ કરીને વીજળી બિલની ગણતરી કરે છે.

જવાબ

દરોનું ટેબલ:

કોડ:

```
units = int(input("Enter consumed units: "))

if units <= 100:
    bill = units * 5.00
elif units <= 200:
    bill = units * 7.50
elif units <= 300:
    bill = units * 10.00
else:
    bill = units * 15.00

print(f"Total Bill: Rs {bill}")
```

સમજૂતી:
- **શરતી તર્ક**: if-elif-else structure
- **દર ગણતરી**: યુનિટ slabs આધારિત
- **યુઝર ઇનપુટ**: interactive billing system

મેમરી ટ્રીક

``Input-Check-Calculate-Display''

પ્રશ્ન પ્રશ્ન 2(અ OR) [03 ગુણ]

આઇડેન્ટિટી ઓપરેટર્સ ઉપર ટૂંક નોંધ લખો.

જવાબ

આઇડેન્ટિટી ઓપરેટર્સ:
મુખ્ય મુદ્દાઓ:
- **is ઓપરેટર**: ઓબ્જેક્ટ identity સરખાવે, values નહીં
- **is not ઓપરેટર**: ઓબ્જેક્ટ્સ જુદા છે કે નહીં ચકાસે
- **મેમરી સરખામણી**: સમાન મેમરી સ્થાન ચકાસે

મેમરી ટ્રીક

``Is-IsNot for object identity''

પ્રશ્ન પ્રશ્ન 2(બ OR) [04 ગુણ]

પાયથોનમાં ઇન્ડેન્ટેશન શું છે? પાયથોનની વિવિધ વિશેષતાઓ સમજાવો.

જવાબ

ઇન્ડેન્ટેશન: કોડ બ્લોક્સ વ્યાખ્યાયિત કરવા માટે લાઇનની શરૂઆતમાં whitespace.
વિશેષતાઓ:

મહત્વ:
- **ઇન્ડેન્ટેશન**: curly braces ને બદલે છે
- **સુસંગત**: સામાન્ય રીતે પ્રતિ level 4 spaces
- **ફરજિયાત**: કોડ માળખું બનાવે છે

મેમરી ટ્રીક

``Simple-Interpreted-Object-Cross-Large''

પ્રશ્ન પ્રશ્ન 2(ક OR) [07 ગુણ]

પાયથોન પ્રોગ્રામ લખો જે નીચેની વિગતોનો ઉપયોગ કરીને વિદ્યાર્થીના વર્ગ/ગ્રેડની ગણતરી કરતો પાયથોન પ્રોગ્રામ લખો.

જવાબ

ગ્રેડિંગ ટેબલ:
કોડ:

```
percentage = float(input("Enter percentage: "))

if percentage >= 70:
    grade = "Distinction"
elif percentage >= 60:
    grade = "First Class"
elif percentage >= 50:
    grade = "Second Class"
elif percentage >= 35:
    grade = "Pass Class"
else:
    grade = "Fail"

print(f"Grade: {grade}")
```

સમજૂતી:
- **બહુવિધ શરતો**: Nested if-elif structure
- **ગ્રેડ નિર્ધારણ**: ટકાવારી ranges આધારિત
- **Float ઇનપુટ**: દશાંશ ટકાવારી handle કરે

મેમરી ટ્રીક

``Distinction-First-Second-Pass-Fail''

પ્રશ્ન પ્રશ્ન 3(અ) [03 ગુણ]

સિલેક્શન કંટ્રોલ સ્ટેટમેન્ટ શું છે? તેની યાદી બનાવો.

જવાબ

સિલેક્શન કંટ્રોલ સ્ટેટમેન્ટ:

મુખ્ય ખ્યાલો:
- **Selection statements**: શરતો આધારે program flow control કરે
- **Boolean evaluation**: True/False logic વાપરે
- **Branching**: execution ના જુદા રસ્તાઓ

મેમરી ટ્રીક

``If-IfElse-IfElif-Nested''

પ્રશ્ન પ્રશ્ન 3(બ) [04 ગુણ]

નેસ્ટેડ લૂપ ઉપર ટૂંક નોંધ લખો.

જવાબ

નેસ્ટેડ લૂપ:
મુખ્ય મુદ્દાઓ:
- **Nested માળખું**: બીજા લૂપની અંદર લૂપ
- **સંપૂર્ણ execution**: આંતરિક લૂપ પૂરું થાય પછી બાહ્ય આગળ વધે
- **Pattern creation**: 2D structures માટે ઉપયોગી

કોડ ઉદાહરણ:

```
for i in range(3):
    for j in range(2):
        print(f"i={i}, j={j}")
```

મેમરી ટ્રીક

``Outer-Inner-Complete-Pattern''

પ્રશ્ન પ્રશ્ન 3(ક) [07 ગુણ]

યુઝર ડિફાઇન ફંક્શન લખો જે 1 થી 100 સુધીની બધી સંખ્યાઓ દર્શાવે, જે 4 થી વિભાજ્ય છે.

જવાબ

કોડ:

```
def display_divisible_by_4():
    print("Numbers divisible by 4 from 1 to 100:")
    for num in range(1, 101):
        if num % 4 == 0:
            print(num, end=" ")
    print()

# Function call
display_divisible_by_4()
```

Return સાથે વિકલ્પ:

```
def get_divisible_by_4():
    return [num for num in range(1, 101) if num % 4 == 0]

```

```
result = get_divisible_by_4()
print(result)
```

મુખ્ય ખ્યાલો:
- **ફંક્શન વ્યાખ્યા**: def keyword નો ઉપયોગ
- **Range ફંક્શન**: 1 થી 100 iteration
- **Modulus ચકાસણી**: num % 4 == 0 શરત
- **List comprehension**: વૈકલ્પિક અભિગમ

મેમરી ટ્રીક

``Define-Range-Check-Display''

પ્રશ્ન પ્રશ્ન 3(અ OR) [03 ગુણ]

રિપીટેશન કંટ્રોલ સ્ટેટમેન્ટ શું છે? તેની યાદી બનાવો.

જવાબ

રિપીટેશન કંટ્રોલ સ્ટેટમેન્ટ:
મુખ્ય ખ્યાલો:
- **Repetition statements**: કોડ બ્લોક્સ વારંવાર execute કરે
- **Iteration control**: looping ની જુદી પદ્ધતિઓ
- **Loop variables**: iteration progress track કરે

મેમરી ટ્રીક

``For-While-Nested''

પ્રશ્ન પ્રશ્ન 3(બ OR) [04 ગુણ]

break અને continue સ્ટેટમેન્ટ વચ્ચેનો તફાવત આપો.

જવાબ

તફાવત:

કોડ ઉદાહરણ:

```
# break example
for i in range(5):
    if i == 3:
        break
    print(i)  # Output: 0, 1, 2

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

મેમરી ટ્રીક

``Break-Exit, Continue-Skip''

પ્રશ્ન પ્રશ્ન 3(ક OR) [07 ગુણ]

યુઝર ડિફાઇન ફંક્શન લખો જે 1 થી 100 સુધીની બધી બેકી સંખ્યાઓ દર્શાવે.

જવાબ

કોડ:

```
def display_even_numbers():
    print("Even numbers from 1 to 100:")
    for num in range(2, 101, 2):
        print(num, end=" ")
    print()

# Alternative method
def display_even_alt():
    even_nums = []
    for num in range(1, 101):
        if num % 2 == 0:
            even_nums.append(num)
    print(even_nums)

# Function call
display_even_numbers()
```

સમજૂતી:

- **કાર્યક્ષમ range**: બેકી સંખ્યાઓ માટે range(2, 101, 2)
- **Modulus પદ્ધતિ**: % 2 == 0 સાથે વૈકલ્પિક ચકાસણી
- **ફંક્શન ડિઝાઇન**: પુનઃઉપયોગી કોડ બ્લોક

મેમરી ટ્રીક

``Range-Step-Even-Display''

પ્રશ્ન પ્રશ્ન 4(અ) [03 ગુણ]

ફંક્શન વ્યાખ્યાયિત કરો. પાયથોનમાં ઉપલબ્ધ વિવિધ પ્રકારના ફંક્શનની યાદી આપો.

જવાબ

ફંક્શન: ખાસ કાર્ય કરતો પુન:ઉપયોગી કોડ બ્લોક.
ફંક્શન પ્રકારો:
ફાયદા:

- **કોડ પુન:ઉપયોગ**: એકવાર લખો, ઘણીવાર વાપરો
- **મોડ્યુલારિટી**: જટિલ સમસ્યાઓને નાના ભાગોમાં વહેંચવી
- **Parameters**: ફંક્શન્સ માટે ઇનપુટ values

મેમરી ટ્રીક

``Built-User-Lambda-Recursive''

પ્રશ્ન પ્રશ્ન 4(બ) [04 ગુણ]

વેરિએબલના સ્કોપ ઉપર ટૂંક નોંધ લખો.

જવાબ

વેરિએબલ સ્કોપ:
કોડ ઉદાહરણ:

```
x = 10  # Global variable

def my_function():
    y = 20  # Local variable
    print(x)  # Access global
    print(y)  # Access local

my_function()
# print(y)  # Error: y not accessible
```

મુખ્ય ખ્યાલો:

- **Variable accessibility**: variables ક્યાં વાપરી શકાય
- **LEGB rule**: Local, Enclosing, Global, Built-in

મેમરી ટ્રીક

``Local-Global-Builtin''

પ્રશ્ન પ્રશ્ન 4(ક) [07 ગુણ]

Python કોડ લખો જે ઉપભોક્તાને મુખ્ય સ્ટ્રિંગ અને સબસ્ટ્રિંગ માટે પૂછે છે અને મુખ્ય સ્ટ્રિંગમાં સબસ્ટ્રિંગની મેમ્બરશિપ તપાસે છે.

જવાબ

કોડ:

```
def check_substring():
    main_string = input("Enter main string: ")
```

```
    substring = input("Enter substring: ")

    if substring in main_string:
        print(f"'{substring}' found in '{main_string}'")
        print(f"Position: {main_string.find(substring)}")
    else:
        print(f"'{substring}' not found in '{main_string}'")

# Enhanced version with case handling
def check_substring_enhanced():
    main_string = input("Enter main string: ")
    substring = input("Enter substring: ")

    if substring.lower() in main_string.lower():
        print("Substring found (case-insensitive)")
    else:
        print("Substring not found")

check_substring()
```

સમજૂતી:

- **યુઝર ઇન્ટરેક્શન**: string collection માટે input()
- **Membership testing**: in operator નો ઉપયોગ
- **Case sensitivity**: વૈકલ્પિક case handling

મેમરી ટ્રીક

``Input-Check-Report-Position''

પ્રશ્ન પ્રશ્ન 4(અ OR) [03 ગુણ]

લોકલ વેરિએબલ અને ગ્લોબલ વેરિએબલ શું છે?

જવાબ

સરખામણી:
ઉદાહરણ:

```
global_var = 100  # Global

def function():
    local_var = 50  # Local
    print(global_var)  # Accessible
    print(local_var)   # Accessible

print(global_var)  # Accessible
# print(local_var)  # Error
```

મેમરી ટ્રીક

``Local-Limited, Global-Everywhere''

પ્રશ્ન પ્રશ્ન 4(બ OR) [04 ગુણ]

પાયથોનના કોઈપણ ચાર બિલ્ટ-ઇન ફંક્શન સમજાવો.

જવાબ

બિલ્ટ-ઇન ફંક્શન:
વધારાના ઉદાહરણો:

```
# len() function
print(len([1, 2, 3, 4]))  # Output: 4

# type() function
print(type(3.14))  # Output: <class 'float'>

# input() function
age = input("Enter age: ")

# print() function
print("Your age is:", age)
```

મેમરી ટ્રીક

``Length-Type-Input-Print''

પ્રશ્ન પ્રશ્ન 4(ક OR) [07 ગુણ]

Python કોડ લખો જે આપેલ સ્ટ્રિંગમાં સબસ્ટ્રિંગને શોધે છે.

જવાબ

કોડ:

```
def locate_substring():
    main_string = input("Enter main string: ")
    substring = input("Enter substring to find: ")

    # Method 1: Using find()
    position = main_string.find(substring)
    if position != -1:
        print(f"Found at index: {position}")
    else:
        print("Substring not found")

    # Method 2: Using index() with exception handling
    try:
        position = main_string.index(substring)
        print(f"Located at index: {position}")
    except ValueError:
        print("Substring not found")

    # Method 3: Find all occurrences
    positions = []
    start = 0
    while True:
        pos = main_string.find(substring, start)
        if pos == -1:
```

```
            break
        positions.append(pos)
        start = pos + 1

    if positions:
        print(f"All positions: {positions}")

locate_substring()
```

મુખ્ય મેથડ:
- **find() method**: index આપે અથવા -1
- **index() method**: index આપે અથવા exception raise કરે
- **બહુવિધ occurrences**: બધી સ્થિતિઓ શોધવા માટે લૂપ

મેમરી ટ્રીક

``Find-Index-Exception-Multiple''

પ્રશ્ન પ્રશ્ન 5(અ) [03 ગુણ]

સ્ટ્રિંગ વ્યાખ્યાયિત કરો. વિવિધ સ્ટ્રિંગ ઓપરેશન્સની યાદી બનાવો.

જવાબ

સ્ટ્રિંગ: quotes માં બંધ characters ની sequence.
ઓપરેશન્સ:
વિશેષતાઓ:
- **Immutable**: સ્ટ્રિંગ બનાવ્યા પછી બદલી શકાતી નથી
- **Indexing**: વ્યક્તિગત characters access કરવું
- **Methods**: manipulation માટે built-in functions

મેમરી ટ્રીક

``Concat-Repeat-Slice-Length-Case''

પ્રશ્ન પ્રશ્ન 5(બ) [04 ગુણ]

આપણે કેવી રીતે ઓળખી શકીએ કે એલિમેન્ટ એ લિસ્ટનો સભ્ય છે કે નહીં? યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

પદ્ધતિઓ:

ઉદાહરણ:

```
fruits = ["apple", "banana", "orange", "mango"]

# Using 'in' operator
if "apple" in fruits:
    print("Apple is available")

# Using 'not in' operator
if "grapes" not in fruits:
    print("Grapes not available")

# Using count() method
count = fruits.count("apple")
if count > 0:
    print(f"Apple found {count} times")
```

મેમરી ટ્રીક

``In-NotIn-Count for membership''

પ્રશ્ન પ્રશ્ન 5(ક) [07 ગુણ]

Python કોડ લખો જે આપેલ સ્ટ્રિંગના બીજા સબસ્ટ્રિંગ સાથે સબસ્ટ્રિંગને બદલે છે. આપેલ સ્ટ્રિંગ 'Welcome to GTU' તરીકે ધ્યાનમાં લો અને સબસ્ટ્રિંગ 'GTU' ને 'Gujarat Technological University' સાથે બદલો.

જવાબ

કોડ:

```
def replace_substring():
    # Given string
    original = "Welcome to GTU"
    old_substring = "GTU"
    new_substring = "Gujarat Technological University"

    # Method 1: Using replace()
    result1 = original.replace(old_substring, new_substring)
    print(f"Original: {original}")
    print(f"Modified: {result1}")

    # Method 2: Manual replacement
    if old_substring in original:
        index = original.find(old_substring)
        result2 = original[:index] + new_substring + original[index + len(old_substring):]
        print(f"Manual method: {result2}")

    # Method 3: Replace all occurrences
    test_string = "GTU offers GTU degree from GTU"
    result3 = test_string.replace("GTU", "Gujarat Technological University")
    print(f"Multiple replacements: {result3}")

replace_substring()
```

આઉટપુટ:

Original: Welcome to GTU
Modified: Welcome to Gujarat Technological University

મુખ્ય મુદ્દાઓ:
- **replace() method**: built-in string function
- **Slicing method**: મેન્યુઅલ string manipulation
- **બધી occurrences**: દરેક instance બદલે છે

મેમરી ટ્રીક

``Find-Replace-Slice-All''

પ્રશ્ન પ્રશ્ન 5(અ OR) [03 ગુણ]

લિસ્ટ વ્યાખ્યાયિત કરો. વિવિધ લિસ્ટ ઓપરેશન્સની યાદી બનાવો.

જવાબ

લિસ્ટ: ક્રમબદ્ધ items નો collection જે modify કરી શકાય છે.
ઓપરેશન્સ:
વિશેષતાઓ:
- **Mutable**: લિસ્ટ બનાવ્યા પછી બદલી શકાય છે
- **Indexed**: સ્થિતિ દ્વારા elements access કરાય છે
- **Dynamic**: કદ વધી અથવા ઘટી શકે છે

મેમરી ટ્રીક

``Add-Remove-Access-Slice-Sort''

પ્રશ્ન પ્રશ્ન 5(બ OR) [04 ગુણ]

સ્ટ્રિંગ સ્લાઇસિંગ ઉપર ટૂંક નોંધ લખો. યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

સ્ટ્રિંગ સ્લાઇસિંગ: [start:end:step] વાપરીને string ના ભાગો extract કરવું.
Syntax:
ઉદાહરણ:

```
text = "Python Programming"

print(text[0:6])   # "Python"
print(text[7:])    # "Programming"
print(text[:6])    # "Python"
print(text[::2])   # "Pto rgamn"
print(text[::-1])  # "gnimmargorP nohtyP"
```

મેમરી ટ્રીક

``Start-End-Step-Reverse''