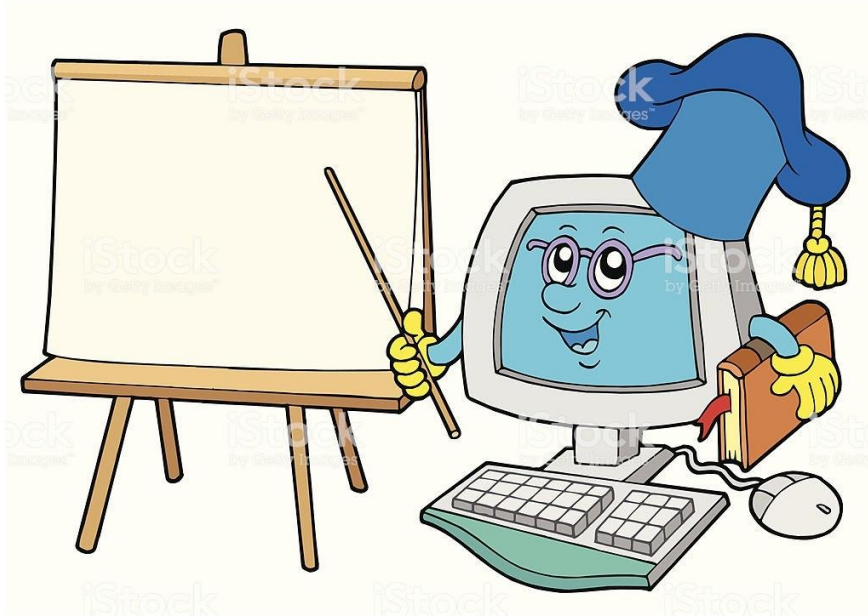




# HIGHER SECONDARY SECOND YEAR COMPUTER SCIENCE

UNIT II – Core Python  
BOOK BACK QUESTION & ANSWERS  
2024 - 25



Prepared By,

**J. KAVITHA, B.Sc, B.Ed, M.C.A, M.Phil.,**  
**Computer Instructor Gr - I**  
GHSS, S.S.KULAM  
Coimbatore – 641107.

<https://kavikalvi.freeweb.co.in/>

## CHAPTER 5: Python - Variables and Operators

### Choose the best answer: (1 Mark)

- Who developed Python?  
(A) Ritche      **B) Guido Van Rossum**      C) Bill Gates      D) Sunder Pitchai
- The Python prompt indicates that Interpreter is ready to accept instruction.  
**(A) >>>**      B) <<<      C) #      D) <<
- Which of the following shortcut is used to create new Python Program?  
(A) Ctrl + C      B) Ctrl + F      C) Ctrl + B      **D) Ctrl + N**
- Which of the following character is used to give comments in Python Program?  
**(A) #**      B) &      C) @      D) \$
- This symbol is used to print more than one item on a single line.  
(A) Semicolon(;)      B) Dollor(\$)      **C) comma(,)**      D) Colon(:)
- Which of the following is not a token?  
**(A) Interpreter**      B) Identifiers      C) Keyword      D) Operators
- Which of the following is not a Keyword in Python?  
(A) break      B) while      C) continue      **D) operators**
- Which operator is also called as Comparative operator?  
(A) Arithmetic      **B) Relational**      C) Logical      D) Assignment
- Which of the following is not Logical operator?  
(A) and      B) or      C) not      **D) Assignment**
- Which operator is also called as Conditional operator?  
**(A) Ternary**      B) Relational      C) Logical      D) Assignment

**Answer the following questions: (2 Marks)**

**1. What are the different modes that can be used to test Python Program ?**

In Python, programs can be written in two ways namely,

- **Interactive mode**                      \* **Script mode.**

**2. Write short notes on Tokens.**

- Python breaks each logical line into a sequence of elementary lexical components known as **Tokens**.
- The normal token types are Identifiers, Keywords, Operators, Delimiters and Literals.

**3. What are the different operators that can be used in Python ?**

- **Operators are special symbols** which represent computations, conditional matching in programming.
- Operators are categorized as Arithmetic, Relational, Logical, Assignment and Conditional.

**4. What is a literal? Explain the types of literal?**

- Literal is a raw data given in a variable or constant.
- In Python, there are various types of literals. They are, **Numeric Literals**  
**String literal** and **Boolean literal**

**5. Write short notes on Exponent data?**

- An Exponent data contains decimal digit part, decimal point, exponent part followed by one or more digits.

**Example:** 12.E04, 24.e04

**Answer the following questions: (3 Marks)**

**1. Write short notes on Arithmetic operator with examples.**

- An arithmetic operator is a mathematical operator used for simple arithmetic.
- It takes two operands and performs a calculation on them.

**Example:** Assume a=100 and b=10. Evaluate the following expressions.

<b>Operator - Operation</b>	<b>Examples</b>	<b>Result</b>
+ (Addition)	>>> a + b	110
- (Subtraction)	>>>a – b	90
* (Multiplication)	>>> a*b	1000
/ (Division)	>>> a / b	10.0
% (Modulus)	>>> a % 30	10
** (Exponent)	>>> a ** 2	10000
// (Floor Division)	>>> a//30 (Integer Division)	3

## 2. What are the assignment operators that can be used in Python?

- In Python, = is a simple assignment operator to assign values to variable.
- There are various compound operators in Python like +=, -=, \*=, /=, %=, \*\*= and //= are also available.

**Example:** Assume x=10

Operator	Example
=	>>> x=10 >>> b="Computer"
+=	>>> x+=20 # x=x+20
-=	>>> x-=5 # x=x-5
*=	>>> x*=5 # x=x*5
/=	>>> x/=2 # x=x/2
%=	>>> x%=3 # x=x%3
**=	>>> x**=2 # x=x**2
//=	>>> x//=3

## 3. Explain Ternary operator with examples.

- Ternary operator is also known as **conditional operator** that evaluates something based on a condition being true or false.
- It simply allows testing a condition in a single line replacing the multiline if-else making the code compact.

**Syntax:** Variable Name = [on\_true] if [Test expression] else [on\_false]

**Example:** min = 50 if 49<50 else 70 # Output: min = 50

## 4. Write short notes on Escape sequences with examples.

- In Python strings, the backslash "\" is a special character, also called the "escape" character. It is used in representing certain whitespace characters:
- "\t" is a tab, "\n" is a newline, and "\r" is a carriage return.

**For example** to print the message "It's raining", the Python command is

```
>>> print ("It\'s raining")
```

It's raining

## 5. What are string literals? Explain.

- In Python a string literal is a **sequence of characters** surrounded by **quotes**.
- Python supports **single, double and triple quotes** for a string.
- A character literal is a **single character** surrounded by **single or double quotes**.
- The value with **triple-quote** ''' ''' is used to give **multi-line** string literal.
- **Example:** strings = "This is Python"  
char = 'C'

## Answer the following questions: (5 Marks)

### 1. Describe in detail the procedure Script mode programming.

- A script is a text file containing the Python statements.
- Once the Python Scripts is created, they are reusable; it can be executed again and again without retyping.

#### (i) Creating Scripts in Python:

- Choose **File** → **New File** or press **Ctrl + N** in Python shell window.
- An **untitled** blank script text editor will be displayed on screen.
- Type the code in Script editor

#### (ii) Saving Python Script:

- Choose **File** → **Save** or Press **Ctrl + S**
- Now, **Save As** dialog box appears on the screen.
- Type the file name with extension **.py** in **File Name** box.
- Then click **Save** button to save your Python script.

#### (iii) Executing Python Script:

- Choose **Run** → **Run Module** or Press **F5**
- If your code has any error, it will be shown in red color in the IDLE window, and Python describes the type of error occurred.
- To correct the errors, go back to Script editor, make corrections, save the file and execute it again.
- For all error free code, the output will appear in the IDLE window of Python.

### 2. Explain input() and print() functions with examples.

#### 1) input() function:

- In Python, **input( )** function is used to accept data as input at run time.  
The syntax for **input()** function is,  
**Variable = input("prompt string")**
- "**Prompt string**" in the syntax is a message to the user, to know what input can be given.
- The **input( )** takes typed data from the keyboard and stores in the given variable.
- If prompt string is not given in **input( )**, the user will not know what is to be typed as input.

**Example:** `>>> city=input ("Enter Your City: ")`

**Output:** Enter Your City: Madurai

#### 2) print() function:

- In Python, the **print()** function is used to display result on the screen.  
**Syntax for print():** **print("String")**  
**print(variable)**
- The **print ( )** displays an entire statement which is specified within **print()**.
- **Comma ( , )** is used as a separator in **print ( )** to print more than one item.

**Example:** `>>> print ("Welcome to Python Programming")`

**Output:** Welcome to Python Programming

### 3. Discuss in detail about Tokens in Python.

- Python breaks each logical line into a sequence of elementary lexical components known as **Tokens**.
- The normal token types are Identifiers, Keywords, Operators, Delimiters and Literals.

#### 1) Identifiers:

- An Identifier is a name used to identify a variable, function, class, module or object.
- An identifier must start with an alphabet (A..Z or a..z) or underscore ( \_ ).
- Identifiers may contain digits (0 .. 9)
- Python identifiers are case sensitive i.e. uppercase and lowercase letters are distinct.
- Identifiers must not be a **python** keyword.

**Example:** Sum, total\_marks, regno, num1

#### 2) Keywords:

- Keywords are special words used by Python interpreter to recognize the structure of program.
- Keywords have **specific meaning for interpreter**; they cannot be used for any other purpose.

**Python Keywords:** false, class, If, elif, else, pass, break etc.

#### 3) Operators:

- **Operators are special symbols** which represent computations, conditional matching in programming.
- Operators are categorized as Arithmetic, Relational, Logical, Assignment and Conditional.

**Example:**  
a=100  
b=10  
print ("The Sum = ",a+b)

**Output:** The Sum = 110

#### 4) Delimiters:

- Python uses the symbols and symbol combinations as delimiters in expressions, lists, dictionaries and strings.

Following are the delimiters: (, ), {, }, [, ], :, ;, +=, \*= ....

#### 5) Literals:

- Literal is a raw data given in a variable or constant.
- In Python, there are various types of literals. They are,
  - **Numeric Literals** consists of digits and are immutable.
  - **String literal** is a sequence of characters surrounded by quotes.
  - **Boolean literal** can have any of the two values: True or False.

## CHAPTER 6: Control Structures

### Choose the best answer: (1 Mark)

- How many important control structures are there in Python?  
**(A) 3**                      B) 4                      C) 5                      D) 6
- elif can be considered to be abbreviation of  
(A) nested if              B) if..else              **C) else if**              D) if..elif
- What plays a vital role in Python programming?  
(A) Statements              B) Control              C) Structure              **D) Indentation**
- Which statement is generally used as a placeholder?  
(A) continue              B) break              **C) pass**              D) goto
- The condition in the if statement should be in the form of  
(A) Arithmetic or Relational expression      B) Arithmetic or Logical expression  
**C) Relational or Logical expression**      D) Arithmetic
- Which is the most comfortable loop?  
(A) do..while              B) while              **C) for**              D) if..elif
- What is the output of the following snippet? `i=1`  
`while True:`  
`if i%3 ==0:`  
`break`  
`print(i,end="")`  
`i +=1`  
**(A) 12**                      B) 123                      C) 1234                      D) 124
- What is the output of the following snippet?  
`T=1`  
`while T:`  
`print(True)`  
`break`  
(A) False                      **B) True**                      C) 0                      D) 1
- Which amongst this is not a jump statement?  
**(A) for**                      B) pass                      C) continue                      D) break
- Which punctuation should be used in the blank? `if <condition>_`  
`statements-block 1`  
`else:`  
`statements-block 2`  
(A) ;                      **B) :**                      C) ::                      D) !

**Answer the following questions: (2 Marks)**

**1. List the control structures in Python.**

Three important control structures are,

- Sequential \* Alternative or Branching \* Iterative or Looping

**2. Write note on break statement.**

- The **break** statement terminates the loop containing it.
- Control of the program flows to the statement immediately after the body of the loop.

**3. Write is the syntax of if..else statement.**

**Syntax:**

```
if <condition>:  
    statements-block 1  
else:  
    statements-block 2
```

**4. Define control structure.**

- A program statement that causes a jump of control from one part of the program to another is called control structure or control statement.

**5. Write note on range () in loop.**

- range() generates a list of values starting from start till stop-1 in for loop.

**The syntax of range() is as follows:**

```
range (start,stop,[step])
```

- Where, **start**– refers to the initial value
- **stop**– refers to the final value
- **step**– refers to increment value, this is optional part.

**Answer the following questions: (3 Marks)**

**1. Write a program to display**

A

A B

A B C

A B C D

A B C D E

**Coding:**

```
a=['A','B','C','D','E']  
for i in range(0,6):  
    for j in range(0,i):  
        print(a[j],end=" ")  
    else:  
        print()
```



## 2. Write note on if..else structure.

- The **if .. else** statement provides control to check the true block as well as the false block.
- **if..else** statement thus provides two possibilities and the condition determines which BLOCK is to be executed.

**Syntax:** if <condition>:

```
statements-block 1
else:
statements-block 2
```

## 3. Using if..else..elif statement write a suitable program to display largest of 3 numbers.

**Coding:**

```
a=int(input("Enter Number 1:"))
b=int(input("Enter Number 2:"))
c=int(input("Enter Number 3:"))
if a>b and a>c:
    print(a,"is biggest")
elif b>a and b>c:
    print(b,"is biggest")
else:
    print(c,"is biggest")
```

**OUTPUT:** Enter Number 1:50

Enter Number 2:14

Enter Number 3:25

50 is biggest

## 4. Write the syntax of while loop.

**Syntax:** while <condition>:

```
statements block 1
[else:
statements block2]
```

## 5. List the differences between break and continue statements.

<b>break</b>	<b>continue</b>
The <b>break</b> statement terminates the loop containing it.	The <b>Continue</b> statement is used to skip the remaining part of a loop
Control of the program flows to the statement immediately after the body of the loop. <b>Syntax:</b> break	Control of the program flows start with next iteration. <b>Syntax:</b> continue

## Answer the following questions: (5 Marks)

### 1. Write a detail note on for loop.

- **for** loop is the most comfortable loop. It is also an entry check loop.
- The condition is checked in the beginning and the body of the loop (statements-block 1) is executed if it is only True otherwise the loop is not executed.

**Syntax:** for counter\_variable in sequence:

```
statements-block 1
[else:      # optional block
statements-block 2]
```

- The counter\_variable is the control variable.
- The sequence refers to the initial, final and increment value.
- **for** loop uses the range()function in the sequence to specify the initial, final and increment values.
- range() generates a list of values starting from start till stop-1 in for loop.

**The syntax of range() is as follows:**

```
range (start,stop,[step])
```

- Where, **start**– refers to the initial value
- **stop**– refers to the final value
- **step**– refers to increment value, this is optional part.

**Example:** for i in range(2,10,2):  
print (i,end=' ')

**Output:** 2 4 6 8

### 2. Write a detail note on if..else..elif statement with suitable example.

- When we need to construct a chain of **if** statement(s) then '**elif**' clause can be used instead of '**else**'.

**Syntax:** if <condition-1>:  
statements-block 1  
elif <condition-2>:  
statements-block 2  
else:  
statements-block n

- In the syntax of **if..elif..else** mentioned above, condition-1 is tested if it is true then statements-block1 is executed.
- Otherwise the control checks condition-2, if it is true statements-block2 is executed and even if it fails statements-block n mentioned in **else** part is executed.

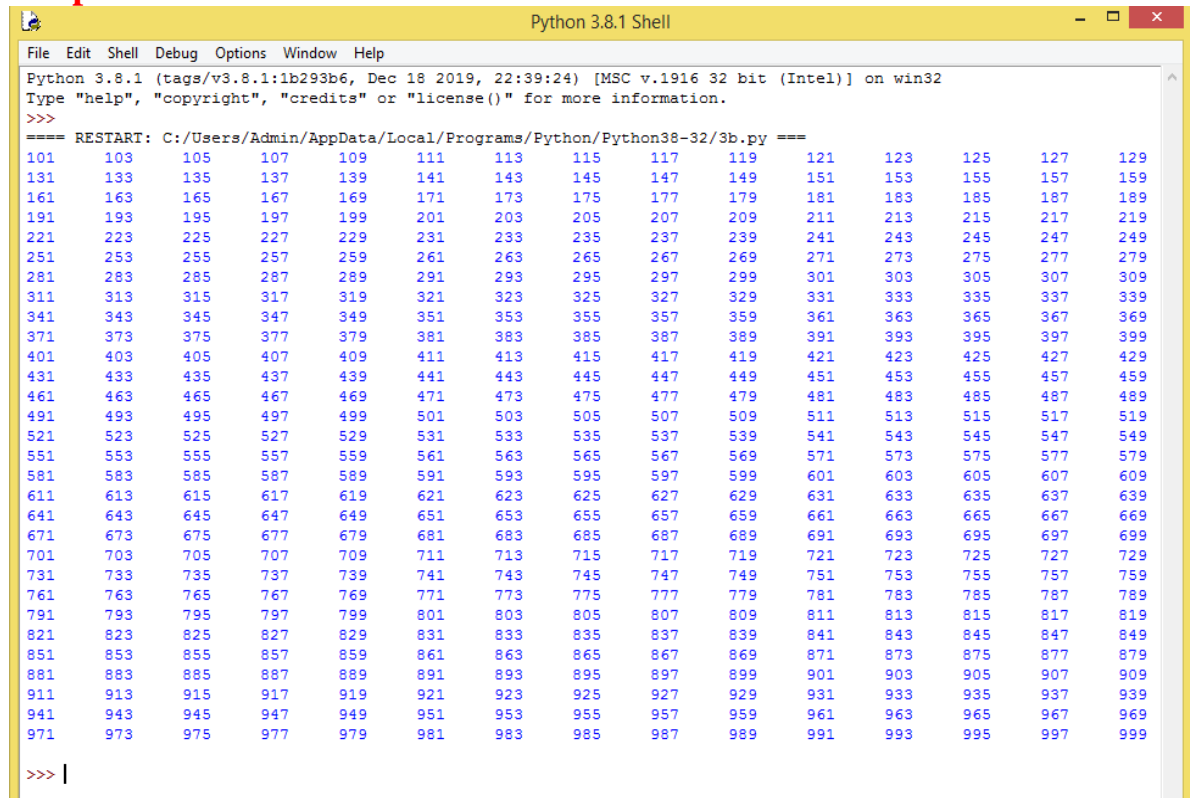
**Example:** m1=int (input("Enter mark in first subject : "))  
m2=int (input("Enter mark in second subject : "))  
avg= (m1+m2)/2  
if avg>=80: print ("Grade : A")  
elif avg>=70 and avg<80: print ("Grade : B")  
elif avg>=60 and avg<70: print ("Grade : C")  
elif avg>=50 and avg<60: print ("Grade : D")  
else: print ("Grade : E")

**Output:** Enter mark in first subject : 34  
Enter mark in second subject : 78  
Grade : D

### 3. Write a program to display all 3 digit odd numbers.

**Coding:**     for i in range(101,1000,2):  
                  print(i,end='\t')

**Output:**



```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python38-32/3b.py =====
101 103 105 107 109 111 113 115 117 119 121 123 125 127 129
131 133 135 137 139 141 143 145 147 149 151 153 155 157 159
161 163 165 167 169 171 173 175 177 179 181 183 185 187 189
191 193 195 197 199 201 203 205 207 209 211 213 215 217 219
221 223 225 227 229 231 233 235 237 239 241 243 245 247 249
251 253 255 257 259 261 263 265 267 269 271 273 275 277 279
281 283 285 287 289 291 293 295 297 299 301 303 305 307 309
311 313 315 317 319 321 323 325 327 329 331 333 335 337 339
341 343 345 347 349 351 353 355 357 359 361 363 365 367 369
371 373 375 377 379 381 383 385 387 389 391 393 395 397 399
401 403 405 407 409 411 413 415 417 419 421 423 425 427 429
431 433 435 437 439 441 443 445 447 449 451 453 455 457 459
461 463 465 467 469 471 473 475 477 479 481 483 485 487 489
491 493 495 497 499 501 503 505 507 509 511 513 515 517 519
521 523 525 527 529 531 533 535 537 539 541 543 545 547 549
551 553 555 557 559 561 563 565 567 569 571 573 575 577 579
581 583 585 587 589 591 593 595 597 599 601 603 605 607 609
611 613 615 617 619 621 623 625 627 629 631 633 635 637 639
641 643 645 647 649 651 653 655 657 659 661 663 665 667 669
671 673 675 677 679 681 683 685 687 689 691 693 695 697 699
701 703 705 707 709 711 713 715 717 719 721 723 725 727 729
731 733 735 737 739 741 743 745 747 749 751 753 755 757 759
761 763 765 767 769 771 773 775 777 779 781 783 785 787 789
791 793 795 797 799 801 803 805 807 809 811 813 815 817 819
821 823 825 827 829 831 833 835 837 839 841 843 845 847 849
851 853 855 857 859 861 863 865 867 869 871 873 875 877 879
881 883 885 887 889 891 893 895 897 899 901 903 905 907 909
911 913 915 917 919 921 923 925 927 929 931 933 935 937 939
941 943 945 947 949 951 953 955 957 959 961 963 965 967 969
971 973 975 977 979 981 983 985 987 989 991 993 995 997 999
>>> |
```

### 4. Write a program to display multiplication table for a given number.

**Coding:**     num=int(input("Display Multiplication Table of "))  
                  for i in range(1,11):  
                      print(i, 'x' ,num, '=' , num\*i)

**Output:**     Display Multiplication Table of 7

```
1 x 7 = 7
2 x 7 = 14
3 x 7 = 21
4 x 7 = 28
5 x 7 = 35
6 x 7 = 42
7 x 7 = 49
8 x 7 = 56
9 x 7 = 63
10 x 7 = 70
```

## CHAPTER 7: Python functions

### Choose the best answer: (1 Mark)

1. A named blocks of code that are designed to do one specific job is called as  
(A) Loop (B) Branching **(C) Function** (D) Block
2. A Function which calls itself is called as  
(A) Built-in **(B) Recursion** (C) Lambda (D) return
3. Which function is called anonymous un-named function  
**(A) Lambda** (B) Recursion (C) Function (D) define
4. Which of the following keyword is used to begin the function block?  
(A) define (B) for (C) finally **(D) def**
5. Which of the following keyword is used to exit a function block?  
(A) define **(B) return** (C) finally (D) def
6. While defining a function which of the following symbol is used.  
(A) ; (semicolon) (B) . (dot) **(C) : (colon)** (D) \$ (dollar)
7. In which arguments the correct positional order is passed to a function?  
**(A) Required** (B) Keyword (C) Default (D) Variable-length
8. Read the following statement and choose the correct statement(s).  
(I) In Python, you don't have to mention the specific data types while defining function.  
(II) Python keywords can be used as function name.  
**(A) I is correct and II is wrong** (B) Both are correct  
(C) I is wrong and II is correct (D) Both are wrong
9. Pick the correct one to execute the given statement successfully.  
if \_\_: print(x, " is a leap year")  
(A)  $x\%2=0$  **(B)  $x\%4==0$**  (C)  $x/4=0$  (D)  $x\%4=0$
10. Which of the following keyword is used to define the function test python(): ?  
(A) define (B) pass **(C) def** (D) while

## Answer the following questions: (2 Marks)

### 1. What is function?

- Functions are named blocks of code that are designed to do one specific job.

### 2. Write the different types of function.

- User-defined Functions
- Lambda Functions
- \* Built-in Functions
- \* Recursion Functions

### 3. What are the main advantages of function?

- It avoids repetition and makes high degree of code reusing.
- It provides better modularity for your application.

### 4. What is meant by scope of variable? Mention its types.

- Scope of variable refers to the part of the program, where it is accessible, i.e., area where you can use it.

**Types:** 1) local scope 2) global scope.

### 5. Define global scope.

- A variable, with global scope can be used anywhere in the program.
- It can be created by defining a variable outside the scope of any function.

### 6. What is base condition in recursive function.

- A recursive function calls itself.
- The condition that is applied in any recursive function is known as base condition.
- A base condition is must in every recursive function otherwise it will continue to execute like an infinite loop.

### 7. How to set the limit for recursive function? Give an example.

- Python stops calling recursive function after 1000 calls by default.
- So, It also allows you to change the limit using `sys.setrecursionlimit(limit_value)`.

**Example:**

```
import sys
sys.setrecursionlimit(3000)
def fact(n):
    if n == 0:
        return 1
    else:
        return n * fact(n-1)
print(fact (2000))
```

**Answer the following questions: (3 Marks)**

**1. Write the rules of local variable.**

- A variable with local scope can be accessed only within the block that it is created in.
- When a variable is created inside the function, the variable becomes local to it.
- A local variable only exists while the function is executing.
- The formal arguments are also local to function.

**2. Write the basic rules for global keyword in python.**

- When we define a variable outside a function, it's global by default. We don't have to use global keyword.
- We use global keyword to read and write a global variable inside a function.
- Use of global keyword outside a function has no effect.

**3. What happens when we modify global variable inside the function?**

- If we modify the global variable inside the function, it will show Unbound Local Error.

**Example : Modifying Global Variable From Inside the Function**

```
c = 1 # global variable
def add():
    c = c + 2 # increment c by 2
    print(c)
add()
```

Output: UnboundLocal Error: local variable 'c' referenced before assignment

**4. Differentiate ceil() and floor() function?**

<b>ceil()</b>	<b>floor()</b>
Returns the smallest integer greater than or equal to x.	Returns the largest integer less than or equal to x.
Syntax: math.ceil (x)	Syntax: math.floor (x)
Ex: >>>import math >>>print(math.ceil(26.7)) Output: 27 >>>Print(math.ceil(-26.7)) Output: -26	Ex: >>>import math >>>print(math.floor(26.7)) Output: 26 >>>Print(math.floor(-26.7)) Output: -27

**5. Write a Python code to check whether a given year is leap year or not.**

```
Coding: n=int(input("Enter the year"))
           if(n%4==0):
               print (n, "is a Leap Year")
           else:
               print (n, "is not a Leap Year")
```

**Output:** Enter the year 2012  
2012 is a Leap Year

## 6. What is composition in functions?

- The value returned by a function may be used as an argument for another function in a nested manner. This is called **composition**.

**For example**, if we wish to take a numeric value as a input from the user, we take the input string from the user using the function **input()** and apply **eval()**function to evaluate its value.

```
>>> n1 = eval(input("Enter an Airthmetic Expression:"))
Enter an Airthmetic Expression:12.0+13.0*2
>>> n1
38.0
```

## 7. How recursive function works?

- Recursive function is called by some external code.
- If the base condition is met then the program gives meaningful output and exits.
- Otherwise, function does some required processing and then calls itself to continue recursion.

**Example:**

```
def fact(n):
    if n == 0:
        return 1
    else:
        return n * fact (n-1)
print (fact (0))
print (fact (5))
```

**Output:**

```
1
120
```

## 8. What are the points to be noted while defining a function?

- Function blocks begin with the keyword “**def**” followed by function name and parenthesis ().
- Any input parameters should be placed within these parentheses.
- The code block always comes after a colon (:) and is indented.
- The statement “**return [expression]**” exits a function, and it is optional. A “**return**” with no arguments is the same as return None.

**Syntax:**

```
def <function_name ([parameter1, parameter2...])> :
    <Block of Statements>
    return <expression / None>
```

## Answer the following questions: (5 Marks)

### 1. Explain the different types of function with an example.

- Functions are named blocks of code that are designed to do one specific job.

#### Types of Functions:

##### User defined Function:

- Functions defined by the users themselves are called user defined function.

**Example:**

```
def hello():
    print ("hello - Python")
    return
```

**Output:** hello – Python

##### Built-in Function:

- Built-in functions are Functions that are inbuilt with in Python.

**Example:** print(), echo() are some built-in function.

##### Lambda Function:

- In Python, anonymous function is a function that is defined without a name.
- While normal functions are defined using the **def** keyword, in Python anonymous functions are defined using the **lambda** keyword.
- Hence, anonymous functions are also called as **lambda** functions.

**Example:**

```
sum = lambda arg1, arg2: arg1 + arg2
print ('The Sum is :', sum(30,40))
print ('The Sum is :', sum(-30,40))
```

**Output:** The Sum is : 70  
The Sum is : 10

##### Recursion Function:

- Functions that calls itself is known as recursive.

**Example:**

```
def fact(n):
    if n == 0:
        return 1
    else:
        return n * fact (n-1)

print (fact (0))
print (fact (5))
```

**Output:** 1  
120



## 2. Explain the scope of variables with an example.

- Scope of variable refers to the part of the program, where it is accessible, i.e., area where you can use it.
- There are two types of scopes: **local scope** and **global scope**.

### Local Scope:

- A variable declared inside the function's body or in the local scope is known as local variable.

### Rules of local variable:

- A variable with local scope can be accessed only within the function/block that it is created in.
- When a variable is created inside the function/block, the variable becomes local to it.
- A local variable only exists while the function is executing. The formal arguments are also local to function.

**Example:**

```
def loc():  
    y=0          # local scope  
    print(y)  
loc()
```

**Output:** 0

### Global Scope:

- A variable, with global scope can be used anywhere in the program.
- It can be created by defining a variable outside the scope of any function/block.

### Rules of global Keyword:

- When we define a variable outside a function, it's global by default. We don't have to use global keyword.
- We use global keyword to read and write a global variable inside a function.
- Use of global keyword outside a function has no effect

**Example:**

```
c = 1          # global variable  
def add():  
    print(c)  
add()
```

**Output:** 1

### 3. Explain the following built-in functions.

(a) id() (b) chr() (c) round() (d) type() (e) pow()

Function	Description	Example
<b>id ()</b>	Return the address of the object in memory.	x=15 print ('address of x is :',id (x)) <b>Output:</b> address of x is : 1357486752
<b>chr ()</b>	Returns the Unicode character for the given ASCII value.	c=65 print(chr(c)) <b>Output:</b> A
<b>round ()</b>	Returns the nearest integer to its input.	x= 17.9 print ( round (x)) <b>Output:</b> 18
<b>type ()</b>	Returns the type of object for the given single object.	x= 15.2 print (type (x)) <b>Output:</b> <class 'float'>
<b>pow ()</b>	Returns the computation of a,b i.e. (a**b ) a raised to the power of b.	a= 5 b= 2 print (pow (a,b)) <b>Output:</b> 25

### 4. Write a Python code to find the L.C.M. of two numbers.

#### Coding:

```
def lcm(x,y):  
    if x>y:  
        greater = x  
    else:  
        greater = y  
    while(True):  
        if((greater % x == 0) and (greater % y == 0)):  
            lcm = greater  
            break  
        greater += 1  
    return lcm  
a = int(input("Enter first number:"))  
b = int(input("Enter second number:"))  
print("LCM is", lcm(a,b))
```

**OUTPUT:** Enter first number: 2  
Enter second number: 3  
LCM is: 6

## 5. Explain recursive function with an example.

- Functions that calls itself is known as recursive.
- Recursion works like loop but sometimes it makes more sense to use recursion than loop.
- Imagine a process would iterate indefinitely if not stopped by some condition is known as infinite iteration.
- The condition that is applied in any recursive function is known as base condition.
- A base condition is must in every recursive function otherwise it will continue to execute like an infinite loop.

### Overview of how recursive function works:

- Recursive function is called by some external code.
- If the base condition is met then the program gives meaningful output and exits.
- Otherwise, function does some required processing and then calls itself to continue recursion.

**Example:**

```
def fact(n):  
    if n == 0:  
        return 1  
    else:  
        return n * fact (n-1)  
print (fact (0))  
print (fact (5))
```

**Output:**

```
1  
120
```

## CHAPTER 8: Strings and String manipulation

### Choose the best answer: (1 Mark)

- Which of the following is the output of the following python code?  

```
str1="TamilNadu"  
print(str1[::-1])
```

(A) Tamilnadu      (B) Tmlau      (C) udanlimaT      **(D) udaNlimaT**
- What will be the output of the following code?  

```
str1 =  
"Chennai Schools"  
str1[7] = "-"
```

(A) Chennai-Schools      (B) Chenna-School  
**(C) Type error**      (D) Chennai
- Which of the following operator is used for concatenation?  
**(A) +**      (B) &      (C) \*      (D) =
- Defining strings within triple quotes allows creating:  
(A) Single line Strings      **(B) Multiline Strings**  
(C) Double line Strings      (D) Multiple Strings
- Strings in python:  
(A) Changeable      (B) Mutable      **(C) Immutable**      (D) flexible
- Which of the following is the slicing operator?  
(A) { }      **(B) []**      (C) < >      (D) ( )
- What is stride?  
(A) index value of slide operation      (B) first argument of slice operation  
(C) second argument of slice operation      **(D) third argument of slice operation**
- Which of the following formatting character is used to print exponential notation in upper case?  
(A) %e      **(B) %E**      (C) %g      (D) %n
- Which of the following is used as placeholders or replacement fields which get replaced alongwith format( ) function?  
**(A) { }**      (B) < >      (C) ++      (D) ^^
- The subscript of a string may be:  
(A) Positive      (B) Negative  
(C) Both (A) and (B)      **(D) Either (A) or (B)**

**Answer the following questions: (2 Marks)**

**1. What is String?**

- String is a data type in python, used to handle array of characters.
- String is a sequence of characters that may be a combination of letters, numbers, or special symbols enclosed within single, double or even triple quotes.

**2. Do you modify a string in Python?**

- Strings in python are immutable. That means, once you define a string modifications or deletion is not allowed.
- However, we can replace the existing string entirely with the new string.

**3. How will you delete a string in Python?**

- Python will not allow deleting a particular character in a string.
- Whereas you can remove entire string variable using **del** command.

**4. What will be the output of the following python code?**

```
str1 = "School"  
print(str1*3)
```

**Output:** School School School

**5. What is slicing?**

- Slice is a substring of a main string.
- A substring can be taken from the original string by using [ ] slicing operator and index or subscript values.
- Using slice operator, we have to slice one or more substrings from a main string.

**Answer the following questions: (3 Marks)**

**1. Write a Python program to display the given pattern**

```
COMPUTER  
COMPUTE  
COMPUT  
COMPU  
COMP  
COM  
CO  
C
```

**Coding:**

```
str="COMPUTER"  
index=len(str)  
for i in str:  
    print(str[:index])  
    index-=1
```

**2. Write a short about the followings with suitable example:**

**(a) capitalize( )      (b) swapcase( )**

<b>FUNCTION</b>	<b>PURPOSE</b>	<b>EXAMPLE</b>
capitalize( )	Used to capitalize the first character of the string	>>> city="chennai" >>> print(city.capitalize()) <b>Output: Chennai</b>
swapcase( )	It will change case of every character to its opposite case vice-versa.	>>> str1="tAmiL NaDu" >>> print(str1.swapcase()) <b>Output:TaMII nAdU</b>

**3. What will be the output of the given python program?**

```
str1 = "welcome"  
str2 = "to school"  
str3=str1[:2]+str2[len(str2)-2:]  
print(str3)
```

**Output: weol**

**4. What is the use of format( )? Give an example.**

- The **format( )** function used with strings is very powerful function used for formatting strings.
- The curly braces { } are used as placeholders or replacement fields which get replaced along with format( ) function.

**Example:**

```
num1=int (input("Number 1: "))  
num2=int (input("Number 2: "))  
print ("The sum of { } and { } is { }".format(num1, num2,(num1+num2)))
```

**Output:**            Number 1: 34  
                      Number 2: 54  
                      The sum of 34 and 54 is 88

**5. Write a note about count( ) function in python.**

- Returns the number of substrings occurs within the given range.
- Remember that substring may be a single character.

**Syntax: count(str, beg, end)**

- Range (beg and end) arguments are optional.

**Example:**        >>> str1="Raja Raja Chozhan"  
                      >>> print(str1.count('Raja'))

**Output: 2**

## Answer the following questions: (5 Marks)

### 1. Explain about string operators in python with suitable example.

Python provides the following string operators to manipulate string.

#### Concatenation (+):

- Joining of two or more strings using plus (+) **operator** is called as **Concatenation**.

**Example:** `>>> "welcome" + "Python"`

**Output:** `'welcomePython'`

#### Append (+ =):

- Adding more strings at the end of an existing string using **operator** += is known as **append**.

**Example:** `>>> str1="Welcome to "  
>>> str1+="Learn Python"  
>>> print (str1)`

**Output:** `Welcome to Learn Python`

#### Repeating (\*):

- The multiplication operator (\*) is used to display a string in multiple number of times.

**Example:** `>>> str1="Welcome "  
>>> print (str1*4)`

**Output:** `Welcome Welcome Welcome Welcome`

#### String slicing([ ]):

- Slice is a substring of a main string.
- A substring can be taken from the original string by using [ ] **slicing operator** and index values.
- Using slice operator, we have to slice one or more substrings from a main string.

**Example:** `>>> str1="THIRUKKURAL "  
>>> print (str1[0])`

**Output:** `T`

#### Stride when slicing string:

- When the slicing operation, we can specify a third argument as the stride, which refers to the number of characters to move forward after the first character is retrieved from the string.
- The default value of stride is 1

**Example:** `>>> str1 = "Welcome to learn Python"  
>>> print (str1[10:16])  
>>> print(str1[::-2])`

**Output:** `Learn  
nhy re teoW`

**Education Is The  
Foundation Of All  
We Do In Life.  
It Shapes Who We Are  
And What We Aspire To Be.**

**ALL THE BEST!**



**J. KAVITHA, B.Sc, B.Ed, M.C.A, M.Phil.,  
Computer Instructor Gr - I  
GHSS, S.S.KULAM  
Coimbatore – 641107.  
☎: 8940762362**