Chhatrapati Shahu Ji Maharaj University, Kanpur

Answer Script Details
Barcode 5457296

Roll No. 23071002365
Total Mark 61/75.00

Exam BACHELOR OF COMPUTER APPLICATIONS_ODD EXA
Subject BCA3001 - PYTHON PROGRAMMING

Question wise Mark Summary

Q.No	Mark	Q.No	Mark	Q.No	Mark	Q.No	Mark
1A	4/5						
1B	4/5						
1C	4/5						
1D	4/5						
1E	4/5						
1F	5/5						
1G	4/5						
1H	4/5						
1I	3/5						
2	13/15						
3	NA/15						
4	NA/15						
5	NA/15						
6	NA/15						
7	NA/15						
8	NA/15						
9	12/15						

Chhatrapati Shahu Ji Maharaj University
Kanpur, Uttar Pradesh

PART-I

Date of Exam: 23-12-24 Room No. 57-13
Paper Code: BCA 3001 Subject: Python Programming
Name of Candidate: Khyati Trivedi Roll No. 2307100236 5

OR Facsimile

Signature of Invigilator

Signature of Candidate: Khyati Trivedi

PART-II

MARKS OBTAINED										
Q.	1	2	3	4	5	6	7	8	9	10
(a)										
(b)										
(c)										
(d)										
(e)										
(f)										
(g)										
(h)										
(i)										
(j)										
Total										
Total Marks in Figures									Max. Marks	
Total Marks in Words										

BCA 3001
Paper Code

Signature of Evaluator

PART-III

Course: Bachelor of Computer Application
Session: 2024-2025 Year/ Semester: III rd Sem
Subject Name: Python Programming
Medium: English ● Hindi ○

Paper Code: BCA 3001

Exam Date: 23 12 2024

Name of Candidate: KHYATI TRIVEDI

Father's Name: S K TRIVEDI

College Code: KN162

Exam Centre Code: KN162

Type of Exam
Regular ● Ex-Student ○
Private ○ Back Paper Exam ○

ANSWER BOOKLET NO.
5457296

BCA3001
Paper Code

Enrollment Number: CSJMA230001295 17

Candidate's Roll Number: 2307100236

Paper Code: 3001

Signature of Candidate: Khyati Trivedi

Signature of Invigilator

C S Facsimile

OR Facsimile

PART-IV

नोट: 1. परीक्षार्थी को निर्देशित किया जाता है कि आवरण पत्रे के पृष्ठ भाग पर अंकित सभी निर्देशों को सावधानी पूर्वक पढ़ें।
2. ओ.एम.आर. में भरी जाने वाली प्रविष्टियाँ काली स्याही से शुरू की जायें। 3. गोलों को काले या नीले बॉलपेन से भरा जायें।

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-I

1. Read the instructions carefully given on the answer script and admit card.
2. Write Date of Exam, Shift, Paper Code & Name of Subject Correctly.
3. Write Name & Roll No. Correctly.
4. Write Semester & Branch Correctly.

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-III

1. Use blue or black ball point pen for writing alphabets & numerals in ☐ boxes.
2. Carefully study the example before you start marking.
3. As shown in the example below, blacken the circles completely.

4. Make no Stray marks on this sheet.
5. **DO NOT WRITE OR MARK ON THE BAR CODE.**

IN ORDER TO AVOD UFM (UNFAIR MEANS) :

1. The Roll No. and Answer Book no. found elsewhere or any other symbol found in the answer book will be treated as unfair means.
2. Any tempering of Bar Code and Booklet no shall be treated as Unfair Means.
3. Do Not bring the materials like slip of paper/mobile/digital diaries/ study material/ revision notes in examination hall. Possession of the mobiles/ digital diaries/electronic/digital/ watch and any other electronic gadget except memory less scientific calculator shall be considered as UFM case.
4. Do not keep or paste currency note in answer script it shall be consider as UFM.

अनुचित साधन से बचने हेतु :

1. उत्तर पुस्तिका के निर्देशित स्थान को छोड़कर अनुक्रमांक एवं उत्तरपुस्तिका का क्रमांक कहीं और न लिखें तथा कोई भी चिन्ह न बनायें क्योंकि यह अनुचित साधन प्रयोग की परिधि में आता है।
2. उत्तर पुस्तिका के बारकोड अथवा उत्तर पुस्तिका संख्या पर छेड़ छाड़ करने पर अनुचित साधन प्रयोग माना जायेगा।
3. परीक्षा कक्ष में निम्न वस्तुएं साथ न लायें, जैसे लिखे हुए कागज के टुकड़ें, मोबाईल, डिजिटल डायरी, डिजिटल घड़ी, घड़ी, पुस्तक यह सभी वस्तुएं जो अनुचित साधन से अन्तर्गत आती है। केवल संबंधित प्रश्नपत्र में ही गैरप्रोग्रामेबल कैलकुलेटर ले जाने की अनुमन्यता होगी।
4. उत्तर पुस्तिकाओं में लगाये न चबे न ही उत्तर पुस्तिका में चिपकायें। ऐसा करना अनुचित साधन प्रयोग की परिधि में आता है।

परीक्षार्थियों को विशेष निर्देश

1. प्रवेश पत्र एवं उत्तर पुस्तिका पर दिये गये निर्देशों को ध्यान से पढ़ें।
2. कवर पृष्ठ के दूसरी तरफ कुछ न लिखें।
3. उत्तर पुस्तिका के पृष्ठों पर दोनों तरफ लिखें।
4. प्रश्न पत्र पर अपने अनुक्रमांक के अतिरिक्त कुछ न लिखें।
5. प्रश्न पत्र कोड एवं प्रश्न पत्र ID सावधानी पूर्वक लिखें।
6. अपनी स्थिति स्पष्ट लिखें।
7. उत्तर पुस्तिका के पृष्ठों की संख्या देखें। अगर उत्तर पुस्तिका में पृष्ठ (1-24) से कम है या फटे हुए है, तो परीक्षा शुरू होने के पूर्व दूसरी उत्तर पुस्तिका ले लें।
8. प्रश्नपत्र को देख, यदि प्रश्नपत्र के विषय कोड, विषय का नाम तथा प्रश्न में कोई त्रुटि है तो उसके परीक्षा शुरू होने से 30 मिनट के अन्दर कक्ष निरीक्षक को सूचित करें, उसके बाद विश्वविद्यालय द्वारा कोई कार्यवाही नहीं की जायेगी।
9. प्रश्नों के उत्तर लिखने के लिये पेंसिल का प्रयोग न करें।
10. की कोरी या अतिरिक्त ग्राफ नहीं दिया जायेगा।

INSTRUCTION TO THE CANDIDATE

1. Read the instructions carefully given on the Question Paper, Admit Card & Answer Script.
2. Do not write anything on back side of the cover page.
3. Write on both sides of pages of answer book.
4. Do not write anything on question paper except Roll Number.
5. Write Paper Code & Question Paper Id carefully.
6. CHECK the number of pages (1-24) or any other kind of damage in your answer script, if found than change the answer script immediately before the commencement of examination.
7. CHECK the Question Paper for any kind of discrepancy e.g. Subject Code, Subject Name, and Question of the Question Paper during first THIRTY MINUTES of the commencement of the exam, so that it can be corrected in TIME. After that no corrections shall be entertained by the university.
8. Do not use pencil for answering the question.
9. Write status correctly e.g. those appearing in carry over papers should fill in status as Carry Over. Those appearing as Ex-Students should fill in status as ex.
10. No supplementary answer book & graph paper will be provided.

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-IV

1. Use blue or black ball point pen for writing alphabets & numerals in ☐ Boxes.
2. Use blue or black ball point pen for filling the circles.

	1	8	1	5	4	3	2	1	6	9
0	○	○	○	○	○	○	○	○	○	○
1	●	○	●	○	○	○	○	●	○	○
2	○	○	○	○	○	○	●	○	○	○
3	○	○	○	○	○	●	○	○	○	○
4	○	○	○	○	●	○	○	○	○	○
5	○	○	○	●	○	○	○	○	○	○
6	○	○	○	○	○	○	○	○	●	○
7	○	○	○	○	○	○	○	○	○	○
8	○	●	○	○	○	○	○	○	○	○
9	○	○	○	○	○	○	○	○	○	●

Note- If your Roll No. is of 10 digits. Please leave first three columns .

Paper Code

1

Section-A

Answer-1

Features of Python

1. High-level language.
   - Python is a High level language which means that it has english like keywords which are understandable by the human.
   - It is later interpreted and converted into machine understandable code which is low level in nature.

2. Object-Oriented language.
   - Python is a object-oriented language which means that everything in python is a object. It follows class-object relation
   - and also follows all oops concepts like.
     Abstraction
     Encapsulation
     Inheritance
     Polymorphism.

3. Dynamically-typed language.
   - Python does not require to declare a variable or function before defining it.
   - The values & objects are assigned to the variables at runtime.

4. Robust Library

- Python is a programming language which has a very Robust library which helps in easy coding.
- It provides various Built-in methods & object that make the work easier & more simply.

5. Portable

- Python is a programming language which can run on all machines irrespective of the operating system.
- We can run the same Python code on windows, Mac Os. if Python shell is installed in it.

6. Free and Open Source

- For using & coding in Python we do not need to buy or pay any amount, we only need to install its shell. that is why it is free.
- The source code of Python is available & all the documentation of source code is public to the users for better Programming.

Answer-2.

Identifiers.

Python Identifiers are the names given by

the programmer to the building block of the program.

- Python Identifiers shall not be Keywords or reserved words because they carry special meaning.
- Rules for naming identifiers -

1. Shall not be a keyword
2. Can only contain alphabets (A-Z), (a-z) & digits (0-9) & an underscore _
3. Must not begin with a digit.
4. Different case are treated different.
5. Shall be of a reasonable length.

Example - valid identifiers

a = " "
first_whole_number = 0
app1 = "Apple"

Invalid identifiers

0ba_b = - X
$app = - X

Keywords.

- Keywords are the special words that convey a special meaning to language compiler.
- Keywords have a special meaning and they tell the Interpreter to perform a task.
- Example - break, continue, if, for.
- We can get a list of all keywords using help > keyword.

- Keywords shall not be used as python identifiers.
- True, False are reserved words but not keywords in python.
- Keywords shall be written exactly how they are defined in python i.e cases shall be considered.

Answer-3.

Features of list in Python.

1. List is a sequence data type in python.
2. It is defined with square brackets [] and elements are separated using a ,.
3. A list is a mutable data type which means it can be modified after it is created.
4. defining a list example.

Syntax - listname = [elem1, elem2 ... elemn]

li = [1, 2, 3, 4]

fruits = ['Apple', 'Tomato', 'grapes']

5. A list can have homogeneous elements within the
   eg: list = ['3001', 'Python', 'BCA'].
6. A list is not a hashable data type.

7. Elements from a list can be accessed through indexes because it is Indexed.
Li=['0','1','2','3']
Li[0] = 0
Li[2] = 1

8. We can perform slicing using slice operator.
Li=['A','B','C','D','E']
Li[0:3] = ['A','B','C']
Here the upper limit is exclusive & lower limit is inclusive.

9. Lists are a ordered data-structure which means that the order of the data in which they were inserted is maintained throughout the program.

10. Python provides various built in functions to perform operations on List. like.
extend(), append(), clear(), copy(), sort(), remove(), pop()

Adding element to a List.
Li = ['1','2','Python','Code']
Li.extend("Papu")
print(list) ['1','2','Python','Code','Papi']
Removing element
Li=['1','2','Code']
Li.remove['2']
print(list) ['1','Code']

11. A list cannot be used as Python dictionary keys because it is mutable & unhashable.

Answer - d.

Docstring.

1. A docstring is the first line of a function, class or module which is enclosed within(""" """) triple quotes)

2. Docstring is used to provide documentation and information about the function, class or module about what is the purpose, aim & work of the object.

3. Syntax -

```
def functionname():
    """A docstring providing
    important information
    about the function"""
    #code
```

ex. def sum(list):
    """This function is use to
    all all elements of a list"""
    return sum(list)

4. We can print the docstring & can have access to the docstring using --doc-- attribute from the 'sys' module.

5. Importance.

1. A docstring is important because it provides necessary information and documentation about the function & object which can be used by another user for better understanding.
2. A docstring is used to have important & necessary info so that the user can have access to it and proceed the further task with better understanding.

6. If we want to include docstring to our program we must always keep in mind to keep it as the first line in the block or scope.

~~4. we can also get the docstring by using help (filename) functi~~

4. A docstring must not include any irrelevant or unnecessary information. It should only contain the most relevant & necessary info dealing with the function or module only.

Answer -e.

1. When we group together the related code within a single python file this is called a module.

2. A module always has .py extention.

3. A module contains related difinition & statements that deal with a similar concept & aspect.

4. Module is a file that focuses on a small job & tries to include all the related tasks & operations related to that job.

5. We can have access to module by importing it into another python file using import keyword. import <module name>

6. A module has its own namespace contains which prevents the naming collision.

module.py

```
def fun():
    return 5
a='module data'
b=0.2
l=['1','2',3]
```

Now this file can be imported in other py file using import

file.py
import module

print(module.a)
print(module.fun())

* The variables or objects of a module can be accessed by a syntax.
import <module name>
<module name.object-name>

8. Importance of Modules

1. Simplicity
A module provides simplicity as all the related & connected data is kept inside a single python file.

2. Maintainability
A module helps to logically organize & maintain the python codes & objects.

3. Scoping
A module helps us a have a separate namespace for all the modules which prevent the namespace collisions.

4. Reusability
When a code is defined in a module it can be used any number of times by any number of files by importing it.
∴ it provides reusability.

Answer-f

```
def countdivisor(n):
    count = 0
    for i in range(1, n):
        if n % i == 0:
            count++
    return count
n = int(input("Enter a number to get its divisor"))
print("The number of divisors of {n}: {countdivisor(n)}")
```

This function counts & prints the number of divisors a a number given by the user.

'Enter a number to get its divisor' - 6
The number of divisors of 6 is 3.

Answer-g.

	Set	Dictionary
1.	A set is a unordered datatype	After Python 3.17 dictionaries are ordered

2. A set is a collection of elements enclosed within {} & separated by , comma.	A dictionary is a collection of key value pair separated by : & each pair is separated using , comma & they are enclosed with {} curly braces.
3. A set can only contain immutable elements	Dictionary value can be any python data-type but keys should always be immutable.
4. We cannot access the elements of set through indexes.	We can access elements through indexing.
5. We can define emty set using set() function.	We can define empty dictionary using {} empty duly braces.
6. ex- set={'1', 2, 'py', 0}	ex- dict={'1': 'Python', '2': 'Programming' '7': 'BCA-300'}
7. We can add elements in set using add() funtion Set={1, 2, 3} Set.add(4) print(Set) {1, 2, 3, 4}	We can add elements in dict using update() funtion. dict = {'1': 'Python'} dict.update({'2': 'Prog'}) print(dict) dict{'1': 'Python', '2': 'Prog'}

Answer-h.

Date-class in python.

- Date class in python is a class of the datetime module of python.
- A date class is a idealized naive class that works on Gregorian time.
- We need a date class to form, manipulate format & work with dates.

Uses-

1. Forming a date-object:

A date object can be formed by using the date constructor & passing it values.

```
Import dat
from datetime import date
d = date(2024, 12, 23)
print(d)
```

Printing today date:

```
from datetime import date
d = datetime.date.today()
print(d)
```

2. For comparison of dates
We can perform comparison b/w dates using >=, >, <, <= operators & it will provide a boolean output.

ex. from datetime import date
d1 = date('2024, 12, 23)
d2 = date(2024, 12, 24)
if d1 < d2:
print('Yes') # Yes because day of d1 is less than day of d2

3. For manipulation of date
We can replace or manipulate the date using the builtin function of python date class. replace()

ex. from datetime import date
d = date(2024, 12, 23)
d = d.replace(day=24)
print(d) # 2024-12-24

4. For Event Scheduling.
Python provides various function for scheduling events & they can be performed later in future time.

5. Get dates from the timestamp.
fromtimestamp(timestamp) function
from datetime import date
d = date.fromtimestamp(2009161)
print(d) # return date from 1 Jan 1970

Answer-15.

Exception.

- Exception are the runtime error that occurs after the execution of the program.
- Though Exception have a traceback but they can be handled.
- Exceptions are the logical error that are encountered at runtime after the program has passed all the syntactical standards.
- Exception are the runtime error that can be handled & solved using the try except framework.
- We can prevent our program from crashing if we keep the exception likely code in the try block.
- If any exception thrown it is catched by the except block and handled properly so that the flow of execution of code does not distort.
- Finally block in try-except-finally is executed no matter what happens error occurs or not.
- When Exception is encounter Exception object is thrown from base class Exception

Paper Code

15

Exception may occur when -

- we try to import a module that does not exist.
- Divide a number by zero.

Types of Exception
Built-in Excepti-
I/O Error - When input output fails.
Index Error When index is out of range for iterable
ZeroDivisionError when we divide a number by zero

Usedefined Exceptions
We can also define user-defined Exception using -

```
class custom(Exception):
    pass
try:
    #code
    throw custom
except custom
    #code.
```

Two built in exception.

1. ZeroDivisionError.
when we try to divide a number by zero which not possible arithmetically.

```
try:
    n = 10/0
except ZeroDivisionError as e
    print(e)
```

Traceback (most recent call last):
ZeroDivisionError: ...

2. IndexError :-
This Error is raised when we try to Access the Index of an iterable that does not fall in the given range.

```
li=["1, 2, 3]
try: for i in range (10):
    print(li[i])
catch IndexError as e:
    print(e)
```

```
# 1
- 2
3
error ---:
Traceback
```

Section - B

Answer-2.

(Continue :)

- Continue is a jump statement in python that is used to perform Irregular & random execution of a python program.
- A continue is a keyword that is used inside the looping constructs to skip that particular iteration and perform the next iteration.
- A continue when Encountered by the python counter it skips the code following the

continue statement & performs & resumes its execution from the next iteration.

- We should always use the continue statement very wisely for proper execution & implementation of our code.

example.

```
for i in range (5):
    if i == 3:
        continue
    else:
        print (i)
```

output

0
1
2
4

The output here has skipped the ~~third~~ fourth iteration & directly performed the fifth iteration.

Break

- Break is also a control jump statement but it is different from continue because -

- Whenever break statement is encountered by the program it skips the present loop block & directly terminates & reaches to the first line after the loop body.

- A break statement is a jump statement that skips the execution of the loop & ~~no~~ does not allow any further iterations to take place.
- example.

```
for i in range (10):
    if i==5:
        break;
    else:
        print (i)
print("end")
```

output.
0
1
2
3
4
end.

Here when the value of i reaches to 5 it sees a condition i==5 which directs the flow of execution is to the end of the ~~progr~~ loop & first statement after the loop body is executed now.

Pass

- Pass is also known as 'Null statement'.
- A pass statement is used when syntactically you require a statement but do not want to include anything for current situation of time.

- A pass statement does nothing but it holds the structure of the program.
- Pass statement does not perform any task it is just a python keyword. which when encounted by the interpreter doesn't produce any error & it also doesn't perform any task.
- The Pass is a placeholder that holds the complete structure of the block & leaves the user to perform the task.

A correct logical syntax can be applied later in place of pass & then it can be executed.

- It helps the user for better programming & understanding.

We can use pass statement in .

1. In loops.

```
for i in range (100):
    pass

print ("performing loops")
```

2. In conditions.

```
n = 10
if n%2 == 0
    pass
print("checked condition")
```

3. In functions

```
def fun():
    pass
fun()
print("function Execution").
```

Program when all three are used.

```
def reciprocal():
    for i in range(2):
        if i == 0:
            continue
        elif i > 10:
            break
        elif i < 0:
            pass # Here we will raise exception later.
        else:
            print(1/i)
reciprocal()
print("Printed reciprocals till 10")
```

Section - C

Try - except

- Try - except is a framework that is used to handle the exception objects that are thrown due to improper logic or algorithm.
- A Try - except structure is a tool which can be used to handle the flow of execution of program even when an exception is encountered.
- It prevents the program flow from randomly exiting & handles the situation which can cause error.
- The Try - except framework has following blocks:
  - Try
  - catch
  - except
  - finally.

Try

- It is a block in which all of the code that is likely to cause an exception is kept.

The try block is always executed.

When no Exception is found the flow of program is transferred to the else block.

If any exception is encountered it is thrown and is handled by the except block.

try:
    # code

except:

- When error is thrown by the try block it is catched & handled using the except.
- It is only executed if Exception is found

try:
    # code
except:
    # 2nd

- else:

It is only executed if the exception is not found.

- finally:

Even if exception is found or not finally block is always executed.

try:
    for i in range

Multiple Exception.

When there are more than one exception then they can be handled using multiple except blocks.

```
def reci(n)
    try:
        if n==0
            raise ZeroDivisionError
        elif n<0
            raise ValueError
        else
            print 1/n

    except ZeroDivisionError as e:
        print(e)
    except ValueError as v:
        print(v)
    else
        print ("no exception")
    finally:
        print ("flow ended")
```

reci(5) # 1/5 0.2 no exception flow ended

reci(-5) ValueError, flow Ended.

reci(0) ZeroDivisionError, flow ended

In this program we are calculating reciprocal. when the value is 0 it raises a ZeroDivision Error.

When the value is less than 0 it raise a value error bcoz it cannot calculate reciproco of -ve numbe.

When a proper the value is encountered, it executed properly in this way. multiple exception are handled carefully

n=0 - ZeroDivisionError
Traceback (ZeroDivisionError)
n<0 - ValueError
Traceback (ValueError)