

AKTU B.Tech Python Programming

BCC302 — Theory Examination 2025–26

Question Paper with Detailed Solutions

Particular	Details
Course	B.Tech
Semester	III Semester
Subject Code	BCC302
Subject	Python Programming
Examination	Theory Examination 2025–26
Time	3 Hours
Maximum Marks	70

SECTION A

Q1. Attempt all questions in brief. 02 × 7 = 14

Q1(a) What are Python variables? Give an example of declaring numeric data types.

Marks: 2

Solution

A variable is a name used to store a value in memory. Python does not require explicit declaration of the variable type; the type is determined from the assigned value.

Example:

```
age = 20
marks = 85.5

print(age)
print(marks)
```

Here, **age** is an integer variable and **marks** is a floating-point variable.

Q1(b) Differentiate between for loop using range and while loop in Python.

Marks: 2

Solution

A **for** loop with `range()` is generally used when the sequence or number of iterations is known. A **while** loop continues as long as its condition remains true.

The main difference is that `range()` supplies the sequence of values for the **for** loop, whereas the **while** loop depends on a condition that must be updated appropriately.

Q1(c) Explain string slicing in Python with an example.

Marks: 2

Solution

String slicing is used to obtain a part of a string. The general syntax is `string[start:stop:step]`. The start position is included and the stop position is excluded.

Example:

```
name = "Python"
print(name[0:3])
```

Output: **Pyt**.

Q1(d) What is a tuple in Python? How is it different from a list?

Marks: 2

Solution

A tuple is an ordered collection of elements. It is immutable, so its elements cannot be changed after creation.

Example:

```
t = (10, 20, 30)
```

A list is mutable and is normally written using square brackets, for example [10, 20, 30]. A tuple is normally written using parentheses.

Q1(e) Define dictionary in Python and give an example of its manipulation.

Marks: 2

Solution

A dictionary is a collection of data stored as **key-value pairs**. It is written using curly brackets.

Example:

```
student = {"name": "Rahul", "marks": 85}
student["marks"] = 90
print(student)
```

The value associated with the **marks** key is changed from 85 to 90.

Q1(f) What are read() and readline() functions in file handling?

Marks: 2

Solution

The read() function reads the contents of a file, either completely or up to a specified number of characters. The readline() function reads one line at a time from the file.

Example:

```
file = open("data.txt", "r")
data = file.read()
print(data)
file.close()
```

Q1(g) Explain encapsulation in Python OOP.

Marks: 2

Solution

Encapsulation is an Object-Oriented Programming concept in which data and the methods that operate on that data are combined inside a class. It also provides controlled access to data.

Example:

```
class Student:
    def __init__(self):
        self.__marks = 90

    def show_marks(self):
        print(self.__marks)

s = Student()
```

```
s.show_marks()
```

Here, `__marks` is treated as a private data member and is accessed through a class method.

SECTION B

Q2. Attempt any three of the following. $07 \times 3 = 21$

Q2(a) Explain Python basic operators and data types. Write a program to check if a number is even or odd using if-else.

Marks: 7

Solution

Python provides different operators for performing operations on data. Important types include arithmetic, relational, logical, assignment, membership and identity operators.

Operator Type	Examples	Purpose
Arithmetic	+, -, *, /, %, //, **	Mathematical operations.
Relational	==, !=, >, <, >=, <=	Comparison of values.
Logical	and, or, not	Combining conditions.
Assignment	=, +=, -=, *=, /=	Assigning and updating values.
Membership	in, not in	Checking membership.
Identity	is, is not	Checking object identity.

Common Python data types: int, float, complex, str, list, tuple, dict, set and bool.

Program:

```
num = int(input("Enter a number: "))

if num % 2 == 0:
    print("The number is even")
else:
    print("The number is odd")
```

Example output: Enter a number: 12 → The number is even.

Q2(b) Describe loop manipulation statements: pass, continue, break. Write a program to print prime numbers between 1 to 50 using for loop.

Marks: 7

Solution

pass: It is a null statement used when a statement is syntactically required but no action is needed.

```
for i in range(5):
    pass
```

continue: It skips the remaining statements of the current iteration and moves to the next iteration.

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```

break: It terminates the loop immediately.

```
for i in range(10):
    if i == 5:
        break
    print(i)
```

Program to print prime numbers from 1 to 50:

```
for num in range(2, 51):
```

```

prime = True

for i in range(2, num):
    if num % i == 0:
        prime = False
        break

if prime:
    print(num)

```

Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47.

Q2(c) Discuss string operations and built-in functions. Write a program to count vowels in a string.

Marks: 7

Solution

A string is a sequence of characters enclosed in single, double or triple quotation marks.

String operations: indexing, slicing, concatenation using +, repetition using * and membership using in or not in.

Function / Method	Purpose
len()	Returns string length.
upper()	Converts characters to uppercase.
lower()	Converts characters to lowercase.
replace()	Replaces part of a string.
find()	Finds the position of a substring.
split()	Splits a string into a list.

Program:

```

text = input("Enter a string: ")
count = 0

for ch in text:
    if ch.lower() in "aeiou":
        count += 1

print("Number of vowels =", count)

```

Example output: For Python Programming, the number of vowels is 4.

Q2(d) Explain list and dictionary built-in functions. Write a program to merge two dictionaries.

Marks: 7

Solution

Important list functions/methods: len() gives the number of elements; append() adds an element; insert() inserts at a position; remove() removes an element; sort() sorts the list; and reverse() reverses it.

Important dictionary functions/methods: len() gives the number of key-value pairs; keys() returns keys; values() returns values; items() returns key-value pairs; update() adds or updates entries; and pop() removes an entry.

Program to merge two dictionaries:

```

dict1 = {"name": "Rahul", "age": 20}
dict2 = {"course": "B.Tech", "marks": 85}

dict1.update(dict2)

```

```
print("Merged Dictionary:")
print(dict1)
```

Output: {'name': 'Rahul', 'age': 20, 'course': 'B.Tech', 'marks': 85}.

Q2(e) Describe file writing functions: write(), writelines(). Write a program to write student records to a file.

Marks: 7

Solution

write(): The write() method writes a string to a file and returns the number of characters written.

```
file = open("data.txt", "w")
file.write("Rahul 85")
file.close()
```

writelines(): The writelines() method writes a sequence of strings to a file. Newline characters must be included when separate lines are required.

```
file = open("data.txt", "w")
lines = ["Rahul 85\n", "Amit 90\n"]
file.writelines(lines)
file.close()
```

Program to write student records:

```
students = [
    "Rahul, 101, 85\n",
    "Amit, 102, 90\n",
    "Priya, 103, 88\n"
]

file = open("students.txt", "w")
file.writelines(students)
file.close()

print("Student records written successfully.")
```

SECTION C

Q3(a) Explain Python blocks and conditional statements. Write a program to grade students based on marks using elif.

Marks: 7

Solution

A block in Python is a group of statements that are treated as one unit. Python uses indentation to define blocks instead of braces.

```
if marks >= 40:
    print("Pass")
    print("Student is successful")
```

Conditional statements are used to execute different blocks depending on conditions. The main forms are if, if-else and if-elif-else.

Program to grade students:

```
marks = int(input("Enter marks: "))

if marks >= 90:
    grade = "A+"
elif marks >= 80:
    grade = "A"
elif marks >= 70:
    grade = "B"
elif marks >= 60:
    grade = "C"
elif marks >= 50:
    grade = "D"
elif marks >= 40:
    grade = "E"
else:
    grade = "F"

print("Grade =", grade)
```

Example output: Enter marks: 85 → Grade = A.

Q3(b) Discuss for loops with strings and lists. Write a program to sum elements of a list.

Marks: 7

Solution

A for loop can iterate through each character of a string.

```
name = "Python"

for ch in name:
    print(ch)
```

A for loop can also iterate through each element of a list.

```
numbers = [10, 20, 30, 40]

for num in numbers:
    print(num)
```

Program to sum list elements:

```
numbers = [10, 20, 30, 40, 50]
total = 0

for num in numbers:
    total = total + num
```

```
print("Sum =", total)
```

Output: Sum = 150.

Q4(a) Explain functions in Python. Write a function to calculate area of a circle.

Marks: 7

Solution

A function is a reusable block of code that performs a specific task. Functions reduce repetition and improve program organization. A function is defined using the def keyword and may accept parameters and return a value.

General form:

```
def function_name(parameters):  
    statements
```

Program:

```
def area_of_circle(radius):  
    area = 3.14 * radius * radius  
    return area  
  
r = float(input("Enter radius: "))  
area = area_of_circle(r)  
print("Area of circle =", area)
```

Example output: Enter radius: 5 → Area of circle = 78.5.

Q4(b) Describe string manipulation methods. Write a program to replace characters in a string.

Marks: 7

Solution

Common string manipulation methods include upper(), lower(), capitalize(), strip(), replace(), split() and find().

Program to replace characters/text:

```
text = "I like Java"  
new_text = text.replace("Java", "Python")  
  
print("Original string:", text)  
print("Modified string:", new_text)
```

Output: Original string: I like Java; Modified string: I like Python.

Q5(a) Explain sets and their operations. Write a program to find difference between two sets.

Marks: 7

Solution

A set is an unordered collection of unique elements. Duplicate elements are not stored in a set.

Operation	Operator	Meaning
Union		All elements from either set.
Intersection	&	Common elements of both sets.
Difference	-	Elements in the first set but not in the second.
Symmetric Difference	^	Elements in either set but not in both.

Program:

```
A = {10, 20, 30, 40, 50}
```

```
B = {30, 40, 50, 60, 70}

difference = A - B

print("Difference between A and B =", difference)
```

Output: Difference between A and B = {10, 20}.

Q5(b) Discuss tuple and list slicing. Write a program to reverse a tuple.

Marks: 7

Solution

Slicing extracts a part of a sequence. The general form is `sequence[start:stop:step]`. The start index is included and the stop index is excluded.

List slicing:

```
numbers = [10, 20, 30, 40, 50]
print(numbers[1:4])
```

Output: [20, 30, 40].

Tuple slicing:

```
data = (10, 20, 30, 40, 50)
print(data[1:4])
```

Output: (20, 30, 40).

Program to reverse a tuple:

```
numbers = (10, 20, 30, 40, 50)
reverse_tuple = numbers[::-1]

print("Original tuple:", numbers)
print("Reversed tuple:", reverse_tuple)
```

Output: Reversed tuple = (50, 40, 30, 20, 10).

Q6(a) Explain manipulating file pointer using seek(). Write a program to read last n lines from a file.

Marks: 7

Solution

A file pointer indicates the current position in a file. The `seek()` method is used to move the file pointer to a specified position.

Syntax:

```
file.seek(position)
```

For example, `seek(0)` moves the pointer to the beginning of the file.

```
file = open("data.txt", "r")
file.seek(5)
data = file.read()
print(data)
file.close()
```

Program to read the last n lines:

```
n = int(input("Enter number of lines: "))

file = open("data.txt", "r")
file.seek(0)
lines = file.readlines()
file.close()

for line in lines[-n:]:
```

```
print(line, end="")
```

Here, seek(0) explicitly positions the file pointer at the beginning before the lines are read. The last n lines are selected using lines[-n:].

Q6(b) Describe regular expressions using re module. Write a program to find all digits in a string.

Marks: 7

Solution

A regular expression is a pattern used to search, match or manipulate text. Python provides the re module for regular-expression operations.

Important functions include re.search(), re.match(), re.findall() and re.sub().

Program to find all digits:

```
import re

text = "My roll number is 125 and my marks are 85."
digits = re.findall(r"\d", text)

print("Digits:", digits)
```

Output: Digits: ['1', '2', '5', '8', '5'].

The regular-expression pattern \d matches a digit.

Q7(a) Explain inheritance and polymorphism. Write a program showing multiple inheritance.

Marks: 7

Solution

Inheritance is an OOP concept in which a child class acquires properties and methods from a parent class. It promotes code reusability.

Polymorphism means many forms. The same operation or method can behave differently depending on the object or data involved.

```
print(10 + 20)
print("Hello " + "Python")
```

In the first statement + performs numeric addition, while in the second it performs string concatenation.

Program showing multiple inheritance:

```
class Father:
    def father_property(self):
        print("Father's property")

class Mother:
    def mother_property(self):
        print("Mother's property")

class Child(Father, Mother):
    def child_property(self):
        print("Child's property")

obj = Child()
obj.father_property()
obj.mother_property()
obj.child_property()
```

Output: Father's property; Mother's property; Child's property.

Q7(b) Discuss Tkinter widgets. Write a program to create a simple window with a label using Tkinter.

Marks: 7

Solution

Tkinter is Python's standard interface for creating graphical user interfaces. It provides widgets such as Label, Button, Entry, Text, Checkbutton, Radiobutton and Listbox.

Widget	Purpose
Label	Displays text or information.
Button	Creates a clickable button.
Entry	Accepts single-line input.
Text	Accepts multi-line text.
Checkbutton	Provides a checkbox option.
Radiobutton	Provides selectable options.
Listbox	Displays a list of selectable items.

Program:

```
import tkinter as tk

window = tk.Tk()
window.title("Python GUI")
window.geometry("400x200")

label = tk.Label(window, text="Welcome to Python Programming")
label.pack()

window.mainloop()
```

The Tk() function creates the main window, Label() creates the label, and mainloop() keeps the GUI application running.