MYCOLLEGEVERSE.IN

BUILDING ACADEMIC IDENTITY · THE STUDENT OS

AKTU · B.TECH · PYTHON PROGRAMMING

AKTU Python Programming – Smart Prep Doc

Curated for B.Tech 2nd Year (CST / CS / IT) using recent 3–4 years question paper patterns.

UNIT-WISE FOCUS PYQ DRIVEN CODE PATTERNS

Use this as your quick-revision "Canva-style" doc – mark topics done, flag tricky questions, and revise only patterns that repeat in AKTU exams.

This Doc Helps You

- Identify high-weightage units (Strings, Dict, Files, NumPy).
- Practice exam-style programs (5 & 10 marks).
- Keep notes clean with MyCollegeVerse branding.

Contents

Quick map of units and what to focus on before your AKTU Python paper.

Unit 3 – Complex Data Types

WEIGHTAGE: ★★★ (MOST ASKED)

SNAPSHOT

- Focus on strings, lists, tuples, sets, dictionaries, and basic operations.
- Most programs are short, pattern-based, and appear as 5 or 10-mark questions.
- If time is less, finishing this unit alone can fetch solid marks.

MOST IMPORTANT QUESTIONS

- Write a Python program to enter marks of five subjects, compute percentage, and display grade.
- Write a Python program to count the number of prime numbers in a given range.
- Write a Python program to print the reverse of a string.
- Given `text = "PythonProgramming"`, perform slicing to: first 6 characters, characters from index 6 to 13, last 5 characters, and a new string formed from first 4 + last 3 characters.
- Create a dictionary for students: name as key and list/tuple of marks as value. Display all elements, compute average for each student, and print the student with highest average.

String & List Patterns

```
# reverse a string
s = input("Enter string: ")
print(s[::-1])

# slicing
text = "PythonProgramming"
print(text[:6])
print(text[6:14])
print(text[-5:])
```

Dictionary & Sets

```
students = {
    "A": [80, 85, 90],
    "B": [70, 75, 80]
}

for name, marks in students.items():
    avg = sum(marks) / len(marks)
    print(name, "->", avg)

# common marks using sets
common = set(students["A"]) & set(students["B"])
print("Common marks:", common)
```

Exam Hints

- Write short comments for multi-step logic.
- Use clear variable names: `marks`, `total`, `percentage`, `grade`.
- Show neat output labels, e.g., `print("Percentage:", percentage)`.

Unit 4 – File & Exception Handling

WEIGHTAGE: ★★ (EVERY YEAR)

SNAPSHOT

- Create, read, write, append, rename, delete files.
- Wrap file operations in `try-except` to manage errors.
- Very common 10-mark "full flow" questions based on one scenario.

TYPICAL EXAM TASKS

- Create `numbers.txt` in a specific directory in read-write mode.
- Check if file exists; write content, append content, then read and display.
- Ask user for filename, read contents, convert to uppercase, and write back.
- Rename file only if it exists, then delete it safely.

File Pattern

```
import os

path = r"E:\files\python\filehandling\numbers.txt"

# create & write
with open(path, "w") as f:
    f.write("10 20 30\n")

# append
with open(path, "a") as f:
    f.write("40 50\n")

# read
with open(path, "r") as f:
    print(f.read())
```

Exception Pattern

```
try:
    name = input("Enter file name: ")
    with open(name, "r") as f:
        data = f.read()
    data = data.upper()
    with open(name, "w") as f:
        f.write(data)
except FileNotFoundError:
    print("File not found!")
except Exception as e:
    print("Error:", e)
```

Unit 5 – NumPy & Scientific Apps

WEIGHTAGE: ★★

SNAPSHOT

- Create 1D & 2D arrays, reshape, slice, and perform matrix multiplication.
- Use math / random modules for small calculators or simulations.
- Expected as at least one full program or short part question.

IMPORTANT PATTERNS

- Create a 1D array of 9 elements and reshape to 3×3.
- Compute sum along rows, columns, and full array.
- Extract a 2×2 subset and multiply matrix with itself.
- Simulate tossing a coin 5 times and compute probability of heads.

NumPy 2D Array

```
import numpy as np

arr = np.arange(1, 10)
mat = arr.reshape(3, 3)

print("Matrix:\n", mat)
print("Row sum:", mat.sum(axis=1))
print("Col sum:", mat.sum(axis=0))
print("Total sum:", mat.sum())

sub = mat[0:2, 0:2]
print("Submatrix:\n", sub)

prod = mat @ mat
print("Product:\n", prod)
```

Coin Toss Probability

```
import random

heads = 0
trials = 5

for _ in range(trials):
    toss = random.randint(0, 1)  # 0-tail, 1-head
    if toss == 1:
        heads += 1

prob = heads / trials
print("Heads:", heads)
print("Probability of head:", prob)
```

How to Use This Doc

Follow a simple flow: practice patterns, tick topics done, and come back only to the parts you found tricky.

SMART STEPS

1. Start with Unit 3 (Strings, Dict, Sets) and solve all listed patterns.
2. Move to Unit 4 (Files & Exceptions) and write full programs once.
3. Finish Unit 5 (NumPy, calculator, simple simulations).
4. Mark questions you found hard, and re-solve them a day before exam.

MYCOLLEGEVERSE RESOURCES

- MyCollegeVerse Notes – clean subject-wise notes for revision.
- MyCollegeVerse.in – explore tools, notes and academic utilities.
- Resume Builder – once exams are done, polish your resume.

Designed as a Canva-style prep doc for AKTU Python Programming