

A PRACTICAL FILE ON

Computer Science Practical File

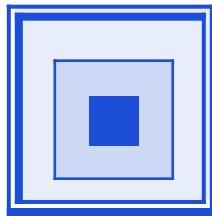
Class XII • Python & SQL practical file

Submitted in partial fulfilment of the practical examination requirements

CLASS XII (CBSE)

in

Class XII - A



SUBMITTED BY

Kabir Malhotra
Roll No. 12A21

CLASS

Class XII - A

Department of Computer Science

SUNRISE PUBLIC SCHOOL, PUNE

Session 2025-26

CERTIFICATE

This is to certify that Kabir Malhotra, a student of Class XII – A, has satisfactorily completed the practical work in Computer Science prescribed by the Central Board of Secondary Education (CBSE) during the academic session 2025–26.

The experiments recorded in this file were performed by the student in the laboratory under my supervision, and the observations and results are the student's own.

Mr. Rohan Desai
Subject Teacher

External Examiner
Date: _____

Mrs. Kavita Iyer
Principal

Acknowledgement

I would like to express my sincere gratitude to my subject teacher, Mr. Rohan Desai, for guidance and encouragement throughout the practical sessions, and for patiently clearing my doubts in the laboratory.

I am also thankful to Mrs. Kavita Iyer, Principal, for providing a well-equipped laboratory, and to the laboratory staff for their help in arranging the apparatus for every experiment.

Finally, I thank my parents and classmates for their constant support while I completed this practical file.

Kabir Malhotra

Index

S. No.	Name of the Experiment	Date	Remarks	Teacher's Sign
1	Fibonacci series and prime check	08 Jul 2025		
2	Counting words, vowels and lines in a text file	22 Jul 2025		
3	Stack implementation using a list	05 Aug 2025		

Write the page number of each experiment in the remarks column once the file is complete.

Experiment 1: Fibonacci series and prime check

Date: 08 Jul 2025

Page No.: _____

Teacher's sign: _____

Aim

To write a Python program that prints the first n terms of the Fibonacci series and reports which of them are prime.

Software / Requirements

- Python 3.10 or later
- IDLE / VS Code

Algorithm

1. Read n from the user.
2. Generate terms by adding the previous two terms.
3. For each term, test divisibility up to its square root.
4. Print the term and whether it is prime.

Program

Python 3

```
def is_prime(n):
    if n < 2:
        return False
    i = 2
    while i * i <= n:
        if n % i == 0:
            return False
        i += 1
    return True

n = int(input("How many terms? "))
a, b = 0, 1
for _ in range(n):
    print(a, "(prime)" if is_prime(a) else "")
    a, b = b, a + b
```

Output

Sample output

```
How many terms? 8
0
1
1
2 (prime)
3 (prime)
5 (prime)
8
13 (prime)
```

Result

Result

The program correctly prints the Fibonacci series and identifies the prime terms.

Points to Note

- Indentation must be consistent.
- Handle $n \leq 0$ gracefully in an improved version.

Experiment 2: Counting words, vowels and lines in a text file

Date: 22 Jul 2025

Page No.: _____

Teacher's sign: _____

Aim

To read a text file and display the number of lines, words and vowels in it.

Software / Requirements

- Python 3
- A text file story.txt in the same folder

Algorithm

5. Open the file in read mode using a with block.
6. Read it line by line.
7. Count lines, split each line into words and count vowels character by character.
8. Display the counts.

Program

```
Python 3
vowels = set("aeiouAEIOU")
lines = words = vcount = 0
with open("story.txt", "r", encoding="utf-8") as f:
    for line in f:
        lines += 1
        words += len(line.split())
        vcount += sum(1 for ch in line if ch in vowels)
print("Lines :", lines)
print("Words :", words)
print("Vowels:", vcount)
```

Output

Sample output

```
Lines : 12
Words : 148
Vowels: 271
```

Result

Result

The program reads the file and reports the number of lines, words and vowels.

Experiment 3: Stack implementation using a list

Date: 05 Aug 2025

Page No.: _____

Teacher's sign: _____

Aim

To implement a stack using a list with push, pop and display operations through a menu.

Software / Requirements

- Python 3

Algorithm

9. Create an empty list to act as the stack.
10. Push appends an element; pop removes the last element if the stack is not empty.
11. Display prints elements from top to bottom.
12. Repeat the menu until the user chooses to exit.

Program

```
Python 3
stack = []
def push(x): stack.append(x)
def pop():
    return stack.pop() if stack else "Underflow"
while True:
    ch = input("1 Push  2 Pop  3 Show  4 Exit: ")
    if ch == "1":
        push(int(input("Value: ")))
    elif ch == "2":
        print("Popped:", pop())
    elif ch == "3":
        print("Stack (top first):", stack[::-1])
    else:
        break
```

Output

Sample output

```
1 Push  2 Pop  3 Show  4 Exit: 1
Value: 10
1 Push  2 Pop  3 Show  4 Exit: 1
Value: 25
1 Push  2 Pop  3 Show  4 Exit: 3
Stack (top first): [25, 10]
1 Push  2 Pop  3 Show  4 Exit: 2
Popped: 25
```

Result

Result

The stack operations work as expected, following the LIFO principle.

Bibliography

- Laboratory manual issued by the department / school.
- Prescribed textbook for the subject (latest edition).
- Class notes and practical demonstrations by the subject teacher.