TEXT FILES — Exam Cheat Sheet

Class XII Computer Science (083) | File Handling | every method, template and trap in one place

1. Opening and closing a file

```
f = open("text.txt", "r")      # filename, mode
...                            # work with f
f.close()                      # ALWAYS close

with open("text.txt", "r") as f:   # preferred: closes automatically
    data = f.read()
```

Mode	Meaning	Pointer starts at	If file does not exist	If file exists
r	read only	beginning	error (FileNotFoundError)	opens it
w	write only	beginning	creates it	ERASES all content
a	append only	end of file	creates it	keeps content, adds at end
r+	read and write	beginning	error	overwrites from start
w+	write and read	beginning	creates it	ERASES all content
a+	append and read	end of file	creates it	keeps content

- Default mode is "r" (text mode). "rt", "wt" mean the same in text mode.
- Text files store data as characters; every line ends with a newline character \n.
- \n is ONE character, even though it is typed as two symbols. It is counted by read() and tell().

2. Writing into a text file

Function	What it does	Returns
f.write(s)	writes the string s; does NOT add \n automatically	number of characters written
f.writelines(L)	writes every string of list L; adds NO \n	None

```
# write mode – file is created fresh every time
f = open("poem.txt", "w")
f.write("Twinkle twinkle little star,\n")
f.write("How I wonder what you are.\n")
f.close()

# writelines with a list (put \n yourself)
L = ["I am learning Python.\n", "This is my first program.\n"]
with open("poem.txt", "w") as f:
    f.writelines(L)

# append mode – old data is kept
f = open("text.txt", "a")
f.write("Python is a cross platform language.\n")
f.close()
```

3. Reading from a text file

Function	What it reads	Returns
f.read()	the whole remaining file	one string
f.read(n)	exactly n characters	string of n characters
f.readline()	one line, including its \n	one string
f.readline(n)	n characters OR up to \n, whichever comes first	one string
f.readlines()	all remaining lines	list of strings, each ending with \n
f.readlines(n)	whole lines until n characters are crossed	list of strings (never a part line)
for line in f:	one line per loop	string per iteration

At end of file read() returns '' (empty string), readline() returns '' and readlines() returns [] — no error is raised.

The three standard reading templates

```
# (a) character by character
```

```
f = open("text.txt", "r")
data = f.read()
for ch in data:
    ...
f.close()

# (b) word by word
f = open("text.txt", "r")
words = f.read().split()      # split() breaks on spaces AND newlines
for w in words:
    ...
f.close()

# (c) line by line
f = open("text.txt", "r")
lines = f.readlines()
for line in lines:
    ...
f.close()
```

4. File pointer — tell() and seek()

- f.tell() returns the current position of the file pointer, counted in characters from the beginning of the file. It moves nothing.
- f.seek(offset, from) moves the file pointer. It reads nothing and returns nothing.

Value of 'from'	Reference point	Example
0 (default)	beginning of the file	`f.seek(10)` or `f.seek(10, 0)`
1	current position	`f.seek(0, 1)`
2	end of the file	`f.seek(0, 2)`

IMPORTANT RULE: in text mode, from = 1 and from = 2 are allowed only when offset is 0. f.seek(5, 1) or f.seek(-10, 2) raises io.UnsupportedOperation. Only from = 0 accepts any offset.

Worked example — file 'text.txt' contains three lines

```
I am learning Python.        # characters 0 to 20,  \n at 21
This is my first file handling program.   # 22 to 60, \n at 61
I am enjoying it.            # characters 62 to 78, \n at 79
# total size of the file = 80 characters

f = open("text.txt", "r")
f.seek(10)          # pointer jumps to character 10
print(f.read(5))    # prints  ing P      pointer now 15
print(f.tell())     # prints  15
f.seek(5, 0)        # back to character 5
print(f.read(5))    # prints  learn      pointer now 10
```

- read(n) consumes n characters starting at the pointer; the new tell() value is old position + n.
- The pointer never moves backwards on its own — only seek() can rewind it.
- After seek(0), text that was already read becomes readable again.

5. Programs that are asked again and again

Count vowels in a file

```
f = open("text.txt", "r")
data = f.read()
c = 0
for ch in data:
    if ch in "AEIOUaeiou":
        c = c + 1
print("Number of vowels =", c)
f.close()
```

Copy a file replacing every vowel with *

```
f1 = open("text.txt", "r")
f2 = open("ABC.txt", "w")
for ch in f1.read():
    if ch in "AEIOUaeiou":
        f2.write("*")
    else:
        f2.write(ch)
```

```
f1.close()
f2.close()
```

Display words shorter than 4 characters

```
f = open("text.txt", "r")
for w in f.read().split():
    if len(w) < 4:
        print(w)
f.close()
```

Count words ending with 'ing'

```
f = open("text.txt", "r")
count = 0
for w in f.read().split():
    if w.endswith("ing"):
        count = count + 1
print(count)
f.close()
```

Count lines starting with 'S' or 's'

```
f = open("text.txt", "r")
count = 0
for line in f.readlines():
    if line[0] == "S" or line[0] == "s":
        count = count + 1
print("Total lines", count)
f.close()
```

Display lines that begin with the word 'You'

```
def showLines():
    f = open("Alpha.txt", "r")
    for line in f.readlines():
        if line.startswith("You"):
            print(line)
    f.close()

showLines()
```

Count the letter 'e' in every line

```
def COUNT():
    F = open("Gratitude.txt")
    T = F.readlines()
    X = 1
    for i in T:
        print("Line", X, ":", i.count("e"))
        X = X + 1
    F.close()

COUNT()
```

Count total lines, words and characters

```
f = open("text.txt", "r")
data = f.read()
print("Characters :", len(data))
print("Words      :", len(data.split()))
print("Lines      :", len(data.split("\n")))
f.close()
```

6. String methods you need with files

Method	Use
`s.split()`	breaks a string into a list of words on spaces and newlines
`s.split("x")`	breaks on the character x
`s.strip()`	removes spaces and \n from both ends (useful after readline)
`s.count("e")`	how many times "e" occurs in s
`s.startswith("You") / s.endswith("ing")`	checks the start / end of a string
`s.upper() / s.lower() / s.isalpha() / s.isdigit()`	case change and character tests
`len(s)`	number of characters (or number of items in a list)

7. Marks are lost here — check every time

- Opening in "w" mode wipes the file. Use "a" when the question says add / append.
- Forgetting f.close() or not using with — always close the file.
- write() does not add \n. If the output needs separate lines you must write \n yourself.
- readline() and readlines() keep the \n, so print() shows an extra blank line. Use print(line, end="") or line.strip().
- After reading the whole file once, a second read gives ''. Use f.seek(0) before reading again.
- In text mode seek() with from = 1 or 2 needs offset 0, otherwise the program stops with an error.
- A function must be called after being defined, otherwise nothing is printed.
- Counting characters: the newline at the end of each line is one character and must be included.