Importance ■■■■■ most asked ■■■□□ often ■□□□□ rarely

Text file programs (3 marks) ■■■■■

Almost every question reads **lines**, **words** or **characters** and tests each one. Learn the three loops; only the `if` changes. All outputs below are for this STORY.TXT:

```
The sun rises in the east.
An apple a day keeps the doctor away.
Python is easy to learn in 2026.
India won the match!
every student must vote.
```

By line: show lines starting with a vowel

```
def show_lines():
    with open('STORY.TXT') as f:
        for line in f:
            if line[0] in 'AEIOUaeiou':
                print(line, end='')
```

```
An apple a day keeps the doctor away.
India won the match!
every student must vote.
```

By word: count the word "the"

```
def count_the():
    c = 0
    with open('STORY.TXT') as f:
        for w in f.read().split():
            if w.lower() == 'the':
                c += 1
    print('Count of the:', c)
```

```
Count of the: 4
```

By character: count vowels, capitals, digits

```
def count_chars():
    v = u = d = 0
    with open('STORY.TXT') as f:
        for ch in f.read():
            if ch in 'AEIOUaeiou':
                v += 1
            if ch.isupper():
                u += 1
            if ch.isdigit():
                d += 1
    print('Vowels:', v, 'Upper:', u, 'Digits:', d)
```

```
Vowels: 41 Upper: 4 Digits: 4
```

IF THE QUESTION ASKS FOR…	USE THIS IF	ASKED
lines starting with 'T'	`line[0] == 'T'`	■■■□
lines not starting with a vowel	`line[0] not in 'AEIOUaeiou'`	■■■□
lines containing the word "vote"	`'vote' in line`	■■■□
lines ending with '!'	`line.rstrip().endswith('!')`	■■■□
count such lines	`c += 1` in the if; print c after the loop	
the longest line	`if len(line) > len(longest): longest = line`	■□□□
words ending with 'te'	`w.endswith('te')`	■■□□
words "Me" or "My"	`w in ('Me', 'My')`	■■■□
words starting with S or H	`w[0] in 'SsHh'`	■■□□
uppercase · lowercase	`ch.isupper()` · `ch.islower()`	■■■■
digits · alphabets	`ch.isdigit()` · `ch.isalpha()`	■■■■

! `split()` keeps punctuation: the last word here is `vote.`, so `w == 'vote'` misses it. Use `w.strip('.,!?')` before comparing. `'vote' in line` also matches "votes"; for whole words, test the stripped words of `line.split()`.

Binary file programs (5 marks) ■■■■■

The usual question: (i) **input a record and append it**, (ii) **update**, show, count or delete the records that match. Each record is a list, here `[ID, Name, Price]`, saved with `pickle`.

(i) Input and append ■■■■■

```
import pickle
def add():
    rec = [int(input('ID: ')), input('Name: '),
           float(input('Price: '))]
    with open('items.dat', 'ab') as f:
        pickle.dump(rec, f)
```

```
ID: 5
Name: Lamp
Price: 650
```

(ii) Show and count the records that match ■■□□□

```
import pickle
def show():
    c = 0
    with open('items.dat', 'rb') as f:
        try:
            while True:
                rec = pickle.load(f)
                if rec[2] > 500:
                    print(rec)
                    c += 1
        except EOFError:
            pass
    print('Found:', c)
```

```
[2, 'Watch', 1500.0]
[3, 'Bag', 800.0]
Found: 2
```

(ii) Update: read all, change, write all back

```
import pickle
def update():
    recs = []
    with open('items.dat', 'rb') as f:
        try:
            while True:
                recs.append(pickle.load(f))
        except EOFError:
            pass
    for r in recs:
        if r[2] > 500:
            r[2] = r[2] - 100
    with open('items.dat', 'wb') as f:
        for r in recs:
            pickle.dump(r, f)
```

The file after `update()`, read back record by record: Watch and Bag are 100 less.

```
[1, 'Pen', 20.0]
[2, 'Watch', 1400.0]
[3, 'Bag', 700.0]
[4, 'Cap', 300.0]
```

VARIATION	CHANGE	ASKED
Delete matching records	in the last loop, dump only `if not` the condition	
Search by a value the user types	`x = input(…)`, then `if r[0] == x`; print "Not found" if none	
Records stored as dictionaries	`{101: ['Asha', 15]}`: use `for k in rec:` and `rec[k]`	

! `pickle.load()` reads **one** record. Loop until it raises **EOFError**. Append uses `'ab'`; writing back uses `'wb'`, which empties the file first, so read everything before you write.

CSV file programs (4 marks)

The usual question: (i) **input a record and add it**, (ii) **show, count or total** the rows that match. Every value read back is a **string**.

(i) Input and add a row

```
import csv
def add():
    row = [input('ID: '), input('Name: '),
           input('Price: ')]
    with open('items.csv', 'a', newline='') as f:
        w = csv.writer(f)
        w.writerow(row)
```

(ii) Show and count rows that match

```
import csv
def search():
    c = 0
    with open('items.csv', 'r', newline='') as f:
        for row in csv.reader(f):
            if float(row[2]) > 1000:
                print(row)
                c += 1
    print('Records:', c)
```

items.csv, then the output:

```
101,Chair,850
102,Table,2400
103,Sofa,15000
```

```
['102', 'Table', '2400']
['103', 'Sofa', '15000']
Records: 2
```

VARIATION	CHANGE	ASKED
Count all records	`c += 1` for every row, no if	
Total (quantity × rate)	`total += int(row[2]) * float(row[3])`	
Name starts with a vowel	`row[1][0] in 'AEIOUaeiou'`	
File has a heading row	`r = csv.reader(f)`, `next(r)`, then loop over `r`	
Write many rows at once	`w.writerows(list_of_rows)`	

! Open with `newline=''`, or Windows puts a blank line between rows. Convert before comparing: `float(row[2]) > 1000`, not `row[2] > 1000`.

File modes

MODE	FOR	FILE MISSING?	EXISTING DATA
`'r'`	read (default)	error	kept
`'w'`	write	created	**erased**
`'a'`	add at the end	created	kept
`'r+'`	read and write	error	kept
`'w+'`	write and read	created	**erased**
`'a+'`	append and read	created	kept

Binary files add **b**: `'rb'`, `'wb'`, `'ab'`, `'rb+'`, `'wb+'`, `'ab+'`. **`with open(…) as f:`** closes the file by itself, even after an error.

Read and write methods, seek and tell

METHOD	RETURNS OR DOES
`read()` · `read(n)`	whole file · next n characters, as one string
`readline()`	next line, with its `\n`
`readlines()`	all lines, as a **list** of strings
`write(s)`	writes a string; no `\n` added
`writelines(L)`	writes a list of strings; no `\n` added
`tell()`	where the file pointer is, in bytes from the start
`seek(n)`	moves the pointer to byte n; `seek(0)` = back to the start

```
f = open('STORY.TXT')
print(f.read(7))
print(f.tell())
f.seek(0)
print(f.readline(), end='')
print(len(f.readlines()))
f.close()
```

```
The sun
7
The sun rises in the east.
4
```

Types of files and paths

	TEXT	BINARY	CSV
Stores	readable characters	bytes, as in memory	text, values split by commas
Open in Notepad	readable	not readable	readable; opens in a spreadsheet
Python uses	`read`, `write`	`pickle`	`csv` module
Extension	.txt	.dat	.csv

Absolute path: from the drive's root, `C:\School\data.txt`.
Relative path: from the current folder, `data.txt` or `..\data.txt`.