1) Write a program to enter the following records in a binary file:

In [2]:

```
# Binary file, so the trick here is to use serialization
```

In [18]:

```
class Item:

    def __init__(self, no, name, qty, price):
        self.no = no
        self.name = name
        self.qty = qty
        self.price = price
        self.amount = self.price * self.qty

    def __str__(self): # in exam create a print_item(item) function, don't do this
        string = '\n'.join([
            "Item No.: " + str(self.no),
            "Item Name: " + self.name,
            "Qty: " + str(self.qty),
            "Price: " + str(self.price),
            "Amount: " + str(self.amount)
        ])
        return string
```

In [19]:

```
import pickle
with open("items.dat", "wb") as f: # wb VVVVV IMMMPPPPP, .dat imp for exam
    item1 = Item(1, "book", 5, 20.2)
    item2 = Item(2, "pen", 7, 5)
    item3 = Item(3, "pencil", 10, 2.5)

    for item in [item1, item2, item3]:
        pickle.dump(item, f)
```

In [20]:

```
n = int(input("No. of records to be read: "))

if n > 3:
    print("So many items not present")

with open("items.dat", "rb") as f: # rb VVVVVVV IIIMPPPP, .dat imp for exam
    for i in range(n):
        item = pickle.load(f)
        amout = item.price * item.qty
        print(item, "\n")
```

```
No. of records to be read: 2
Item No.: 1
Item Name: book
Qty: 5
Price: 20.2
Amount: 101.0

Item No.: 2
Item Name: pen
Qty: 7
Price: 5
Amount: 35
```

2) Write a program to accept string/sentences from the user till the user enters “END” to. Save the data in a text file and then display only those sentences which begin with an uppercase alphabet

In [17]:

```
with open("sentences.txt", "w") as f:

    sentence = input("Enter sentence: ")
    while sentence != "END":
        f.write(sentence + "\n")
        sentence = input("Enter sentence: ")

with open("sentences.txt", "r") as f:
    print("Output: ")
    for sentence in f:
        if sentence[0].isupper():
            print(sentence, end="") # end="" coz the sentence already has \n
```

```
Enter sentence: aldkfja
Enter sentence: Adkafj
Enter sentence: laksjdf
Enter sentence: DLKJAF
Enter sentence: alkdjf
Enter sentence: END
Output:
Adkafj
DLKJAF
```

3) Write the file mode that will be used for opening the following files. Also, write the Python statements to open the following files:

a text file “example.txt” in both read and write mode

In [21]:

```
f = open("./4 Unit 1 - Assignment files/example.txt", "w+")
print("no error")
# w+ read and write both and create file if not exist
f.close()
```

```
no error
```

a binary file “bfile.dat” in write mode

In [22]:

```
f = open("./4 Unit 1 - Assignment files/bfile.dat", "wb")
print("no error")
f.close()
```

```
no error
```

a text file “try.txt” in append and read mode

In [24]:

```
f = open("./4 Unit 1 - Assignment files/try.txt", "ab+") # a+b or ab+
print("no error")
# a+b or ab+ read and append both and create file if not exist
f.close()
```

```
no error
```

a binary file “btry.dat” in read only mode.

In [29]:

```
# FILE SHOULD BE CREATED PHYSICALLY FIRST
# f = open("./4 Unit 1 - Assignment files/btry.dat", "w")
# print("no error")
# f.close()

f = open("./4 Unit 1 - Assignment files/btry.dat", "rb")
print("no error")
f.close()
```

```
no error
```

4) Implement a program that accepts filename as an input from the user. Open the file and count the number of times a character appears in the file and display the occurrence in a dictionary.

In [1]:

```
filename = input("Enter file name: ")
f = open(filename, "r")

string = f.read()
# char_count = {}

# for char in string:
#     if char in char_count:
#         char_count[char] += 1
#     else:
#         char_count[char] = 0

# print(char_count)

# or use set and dict together to create Counter

from collections import Counter
char_counts = Counter(string)
print(char_counts)

f.close()
```

```
Enter file name: lines.txt
Counter({' ': 495, 'e': 306, 'i': 285, 's': 258, 'a': 247, 'u': 238, 't': 22
6, 'n': 180, 'l': 157, 'r': 142, 'm': 129, 'c': 120, 'o': 105, 'd': 72, 'p':
67, '.': 65, 'b': 37, ',': 34, 'f': 34, 'q': 34, 'g': 32, 'v': 32, 'h': 17,
'P': 9, 'D': 9, 'A': 8, 'M': 8, '\n': 8, 'C': 7, 'S': 5, 'N': 5, 'x': 5,
'I': 3, 'j': 3, 'F': 3, 'U': 3, 'V': 2, 'E': 2})
```

5) Implement a python program to validate a mobile number. The number should start with 7, 8 or 9 followed by 9 digits.

In [32]:

```
import re

n = input("Enter 10 digit mobile number: ")
if re.match(r'[789]\d{9}', n):
    print("Number is correct")
else:
    print("Number is not correct")
```

```
Enter 10 digit mobile number: 7356837843
Number is correct
```

In [ ]:

```

```