Viết chương trình tìm nghiệm của phương trình LotKa Volterro

Bài số 9: Tập hợp trong python - Python sets

In [ ]:
```
# Bài 1: Viết chương trình nhập vào một danh sách số nguyên, loại bỏ các số trùn
# set.

input_str = input("Nhập danh sách số nguyên cách nhau dấu phẩy: ")

num_list = []
num_list = [int(x.strip()) for x in input_str.split(',')]

num_set = set(num_list)

print("Danh sách ban đầu: ", num_list)
print("Danh sách loại bỏ trùng lặp: ", num_set)
```

```
Nhập danh sách số nguyên cách nhau dấu phẩy: 4, 5, 6, 7,5
Danh sách ban đầu:  [4, 5, 6, 7, 5]
Danh sách loại bỏ trùng lặp:  {4, 5, 6, 7}
```

In [ ]:
```
# Bài 2: Viết chương trình nhập hai set từ bàn phím, in ra phần tử chung của chú
set_A = input("Nhập set A (cách nhau dấu cách): ").split()
set_B = input("Nhập set B (cách nhau dấu cách): ").split()

set_A = set(s.strip() for s in set_A)
set_B = set(s.strip() for s in set_B)

ptu_giao = set_A & set_B
print(ptu_giao)
```

```
Nhập set A (cách nhau dấu cách): 34 5 6
Nhập set B (cách nhau dấu cách): 4 5 7 8
{'5'}
```

In [25]:
```
# Bài 3: Viết chương trình nhập hai set từ bàn phím, in ra hợp của chúng.
set_A = input("Nhập set A (cách nhau dấu cách): ").split()
set_B = input("Nhập set B (cách nhau dấu cách): ").split()

set_A = set(s.strip() for s in set_A)
set_B = set(s.strip() for s in set_B)

ptu_hop = set_A | set_B
print(ptu_hop)
```

```
Nhập set A (cách nhau dấu cách): 3 4 2 6 4
Nhập set B (cách nhau dấu cách): 12 3 1 5 6
{'12', '6', '5', '3', '2', '1', '4'}
```

In [26]:
```
# Bài 4: Viết chương trình tạo set chứa các số chẵn từ 1 đến 20.
even_set = {x for x in range(1, 21) if x%2 == 0}
print("Set chứa các số chẵn từ 1 đến 20: ", even_set)
```

```
Set chứa các số chẵn từ 1 đến 20:  {2, 4, 6, 8, 10, 12, 14, 16, 18, 20}
```

In [27]:
```
# Bài 5: Viết chương trình tạo set chứa các số lẻ từ 1 đến 20.
# Bài 4: Viết chương trình tạo set chứa các số chẵn từ 1 đến 20.
odd_set = {x for x in range(1, 21) if x%2 != 0}
print("Set chứa các số lẻ từ 1 đến 20: ", odd_set)
```

```
Set chứa các số lẻ từ 1 đến 20:  {1, 3, 5, 7, 9, 11, 13, 15, 17, 19}
```

In [28]:
```
# Bài 6: Viết chương trình kiểm tra xem một chuỗi có chứa tất cả ký tự trong một
# trước hay không.
set_ky_tu = {'p', 'i'}
chuoi = "day la pi"
set_chuoi = set(chuoi)
check = set_ky_tu <= set_chuoi
print(check)
```

```
True
```

In [29]:
```
# Bài 7: Viết chương trình kiểm tra một từ có chứa ký tự trùng lặp không (dùng s
word = input("Nhập từ để kiểm tra: ")
set_word = set(word)
if len(set_word) == len(word):
  print("Không có ký tự trùng lặp")
else:
  print("Có ký tự trùng lặp")

print("Từ đã nhập: ", word)
print("Set: ", set_word)
```

```
Nhập từ để kiểm tra: hong yen
Có ký tự trùng lặp
Từ đã nhập:  hong yen
Set:  {'e', ' ', 'y', 'g', 'h', 'n', 'o'}
```

In [30]:
```
# Bài 8: Viết chương trình tìm ra các ký tự chung giữa hai chuỗi.
string_A = input("Nhập chuỗi A: ")
string_B = input("Nhập chuỗi B: ")

set_A = set(string_A)
set_B = set(string_B)

kytu_chung = set_A & set_B
print("Ký tự chung: ", kytu_chung)
```

```
Nhập chuỗi A: xinchaopython
Nhập chuỗi B: xinchao
Ký tự chung:  {'i', 'x', 'h', 'n', 'o', 'a', 'c'}
```

In [32]:
```
# Bài 9: Viết chương trình nhập vào một câu, in ra tất cả các từ khác nhau trong
# set).
cau = input("Nhập vào câu: ")
list_tu = cau.lower().split()
print(list_tu)
set_tu = set(list_tu)
print(set_tu)
print("Danh sach tu khac nhau: ", set_tu)
```

```
Nhập vào câu: Tấm và Cám là hai chị em cùng cha khác mẹ. Tấm vốn hiền lành, xinh đẹp lại chăm chỉ.
['tấm', 'và', 'cám', 'là', 'hai', 'chị', 'em', 'cùng', 'cha', 'khác', 'mẹ.', 'tấm', 'vốn', 'hiền', 'lành,', 'xinh', 'đẹp', 'lại', 'chăm', 'chỉ.']
{'hai', 'cám', 'xinh', 'em', 'hiền', 'lành,', 'tấm', 'đẹp', 'và', 'là', 'chị', 'mẹ.', 'vốn', 'khác', 'cha', 'cùng', 'chỉ.', 'lại', 'chăm'}
Danh sach tu khac nhau:  {'hai', 'cám', 'xinh', 'em', 'hiền', 'lành,', 'tấm', 'đẹp', 'và', 'là', 'chị', 'mẹ.', 'vốn', 'khác', 'cha', 'cùng', 'chỉ.', 'lại', 'chăm'}
```

In [33]:
```python
# Bài 10: Viết chương trình tính số từ duy nhất trong một đoạn văn bản.
doan = input("Nhập vào đoạn văn bản: ")
list_tu = doan.lower().split()

set_tu = set()
for tu in list_tu:
    cleaned_tu = tu.strip(',.!?"')
    set_tu.add(cleaned_tu)

print("Tổng số từ: ", len(list_tu))
print("Danh sach tu khac nhau: ", set_tu)
print("Tổng số từ duy nhất: ", len(set_tu))
```

```
Nhập vào đoạn văn bản: Tấm và Cám là hai chị em cùng cha khác mẹ. Tấm vốn hiền lành, xinh đẹp lại chăm chỉ. Vốn mồ côi cha mẹ từ sớm, Tấm phải sống cùng dì ghẻ và Cám. Mọi công việc trong nhà đều đến tay nhưng vẫn bị mẹ con Cám ngược đãi, tìm cách hãm hại.
Tổng số từ:  55
Danh sach tu khac nhau:  {'hai', 'sớm', 'việc', 'nhà', 'bị', 'con', 'cám', 'hại', 'xinh', 'chỉ', 'em', 'hiền', 'phải', 'đến', 'ngược', 'từ', 'tấm', 'mồ', 'vẫn', 'mẹ', 'đẹp', 'cách', 'và', 'nhưng', 'dì', 'là', 'chị', 'mọi', 'côi', 'trong', 'đều', 'hãm', 'lành', 'vốn', 'khác', 'sống', 'cha', 'công', 'đãi', 'cùng', 'tìm', 'lại', 'tay', 'ghẻ', 'chăm'}
Tổng số từ duy nhất:  45
```

In [34]:
```python
# Bài 11: Viết chương trình tìm tất cả các phần tử xuất hiện trong ít nhất 2 tro
# cho trước.
taphop_1 = {1, 2, 3, 4}
taphop_2 = {1, 6, 8, 9}
taphop_3 = {2, 6, 4, 5}

tap_giao = (taphop_1 & taphop_2) | (taphop_2 & taphop_3) | (taphop_3 & taphop_1)
print("Các phần tử xuất hiện trong ít nhất 2 tập cho trước là: ", tap_giao)
```

```
Các phần tử xuất hiện trong ít nhất 2 tập cho trước là:  {1, 2, 4, 6}
```

In [35]:
```python
# Bài 12: Viết chương trình so sánh 2 set và in ra phần tử chỉ có trong tập thứ
set_A = {1, 2, 3, 4}
set_B = {4, 3, 6, 7}

set_hieu = set_A - set_B
print("Phần tử chỉ có trong tập thứ nhất là: ", set_hieu)
```

```
Phần tử chỉ có trong tập thứ nhất là:  {1, 2}
```

In [36]:
```python
# Bài 13: Viết chương trình so sánh 2 set và in ra phần tử chỉ có trong tập thứ
set_A = {1, 2, 3, 4}
set_B = {4, 3, 6, 7}

set_hieu = set_B - set_A
print("Phần tử chỉ có trong tập thứ hai là: ", set_hieu)
```

```
Phần tử chỉ có trong tập thứ hai là:  {6, 7}
```

In [37]:
```
# Bài 14: Viết chương trình nhập vào danh sách số nguyên, tìm ra số khác nhau và
# tăng dần.
so_nguyen = input("Nhập vào danh sách số nguyên (cách nhau bởi dấu cách): ")
num_list = []
num_list = [int(x.strip()) for x in so_nguyen.split()]
num_set = set(num_list)
print("Danh sách số duy nhất: ", num_set)
num_set_sort = sorted(num_set)
print("Danh sách số khác nhau sắp xếp tăng dần: ", num_set_sort)
```

```
Nhập vào danh sách số nguyên (cách nhau bởi dấu cách): 4 3 7 8 6 5 90 23 12
Danh sách số duy nhất:  {3, 4, 5, 6, 7, 8, 12, 23, 90}
Danh sách số khác nhau sắp xếp tăng dần:  [3, 4, 5, 6, 7, 8, 12, 23, 90]
```

In [38]:
```
# Bài 15: Viết chương trình loại bỏ tất cả các ký tự đặc biệt trong chuỗi bằng c
s = "Xin**&%chao, day(^&*la!@#$%^pi)"
ky_tu_dac_biet = "!@#$%^&*()_+=,./;'[]]\<>?:{}\|"
chuoi = "".join(char for char in s if char not in ky_tu_dac_biet)
print("Chuỗi sau khi lọc: ", chuoi)
```

```
Chuỗi sau khi lọc:  Xinchao daylapi
<>:3: SyntaxWarning: invalid escape sequence '\<'
<>:3: SyntaxWarning: invalid escape sequence '\<'
/tmp/ipython-input-1052743614.py:3: SyntaxWarning: invalid escape sequence '\<'
  ky_tu_dac_biet = "!@#$%^&*()_+=,./;'[]]\<>?:{}\|"
```

In [40]:
```
# Bài 16: Viết chương trình tìm tất cả ký tự có trong chuỗi A nhưng không có tro

str_A = input("Nhập A: ")
str_B = input("Nhập B: ")

set_A = set(str_A)
set_B = set(str_B)

set_hieu = set_A - set_B

print(set_hieu)
```

```
Nhập A: xinchaoban
Nhập B: xinbao
{'c', 'h'}
```

In [41]:
```
# Bài 17: Viết chương trình tìm tất cả ký tự xuất hiện trong cả chuỗi A và chuỗi

str_A = input("Nhập A: ")
str_B = input("Nhập B: ")

set_A = set(str_A)
set_B = set(str_B)

set_hop = set_A | set_B

print(set_hop)
```

```
Nhập A: xinchao
Nhập B: xinchaoban
{'i', 'b', 'x', 'h', 'n', 'o', 'a', 'c'}
```

In [71]:
```
# Bài 18: Viết chương trình trộn 2 danh sách rồi loại bỏ phần tử trùng nhau bằng
list1= "xinchao"
list2= "python"

combined_list = list1 + list2

my_set = set(combined_list)
print(my_set)
```

```
{'i', 't', 'p', 'x', 'y', 'h', 'n', 'o', 'a', 'c'}
```

In [45]:
```
# Bài 19: Viết chương trình nhập vào một danh sách tên, in ra số tên khác nhau.
s_name = input("Nhập vào danh sách tên cách nhau dấu phẩy: ")
list_name = s_name.split(',')
set_name = set()
for name in list_name:
  unique_name = name.strip()
  set_name.add(unique_name)
print("Danh sách tên ban đầu: ", list_name)
print("Danh sách tên khác nhau: ", set_name)
```

```
Nhập vào danh sách tên cách nhau dấu phẩy: bình, an, mai, huy, hải, hiền, lan, ma
i, huy
Danh sách tên ban đầu:  ['bình', ' an', ' mai', ' huy', ' hải', ' hiền', ' lan',
' mai', ' huy']
Danh sách tên khác nhau:  {'huy', 'hải', 'an', 'mai', 'bình', 'hiền', 'lan'}
```

In [46]:
```
# Bài 20: Viết chương trình kiểm tra xem một set có phải là tập con thực sự của
# khác không.
set_father = "xinchaopythondaylapi"
set_child = "xinchaohaha"
if (set_child <= set_father):
  print("set_child là set con của set_father")
else:
  print("set_child không là set con của set_father")
```

```
set_child là set con của set_father
```

Bài 12: Kiểu dữ liệu Từ điển - Python Dictionaries

Nhóm 2: Vận dụng (20 câu)

In [47]:
```
dic = {
    "pi": 15,
    "haha":15,
    "tute": 20,
    "chip": 20
}
print(dic)
```

```
{'pi': 15, 'haha': 15, 'tute': 20, 'chip': 20}
```

In [48]:
```
student_info = dict(name = "pi", age = 21, major = "DS")
print(student_info)
print(student_info["name"])
print(student_info.get("age"))
```

```
student_info["age"] = 23
student_info["graduation_year"]= 2025
print(student_info)

del student_info["major"]
grad = student_info.pop("graduation_year")
print(student_info)
print("Removed: ", grad)
```

```
{'name': 'pi', 'age': 21, 'major': 'DS'}
pi
21
{'name': 'pi', 'age': 23, 'major': 'DS', 'graduation_year': 2025}
{'name': 'pi', 'age': 23}
Removed:  2025
```

In [49]:
```
student = {"name": "Alice", "age": 22, "major": "CS"}

for k in student:
  print("Key: ", k, "Value: ", student[k])

for v in student.values():
  print("Value: ", v)

for k, v in student.items():
  print(f"{k}: {v}")
```

```
Key:  name Value:  Alice
Key:  age Value:  22
Key:  major Value:  CS
Value:  Alice
Value:  22
Value:  CS
name: Alice
age: 22
major: CS
```

In [50]:
```
d = {"a":1, "b":2}
print(d.keys())
print(d.values())
print(d.items())
print(d.get("a"))
print(d.get("z", 0))
d.update({"c":3})
print(d)
d2 = d.copy()
print(d2)
d.clear()
print(d)
```

```
dict_keys(['a', 'b'])
dict_values([1, 2])
dict_items([('a', 1), ('b', 2)])
1
0
{'a': 1, 'b': 2, 'c': 3}
{'a': 1, 'b': 2, 'c': 3}
{}
```

In [51]:
```
keys = ["a", "b", "c"]
d = dict.fromkeys(keys, 0)
print(d)

student = {"name":"Alice"}
print(student.setdefault("age", 21))
print(student)
```

```
{'a': 0, 'b': 0, 'c': 0}
21
{'name': 'Alice', 'age': 21}
```

In [52]:
```
d1 = {"a": 1, "b": 2}
d2 = {"c": 3, "d": 4}

d3 = d1 | d2
print(d3)

d1 |= d2
print(d1)
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

In [53]:
```
# Bài 1: Viết chương trình đếm số lần xuất hiện của từng từ trong một chuỗi, lưu
# vào dictionary.
text = "Python for Data Science is great. Python is powerful. "
import re
words = re.sub(r'[^\w\s]', '',text.lower()).split()
print(words)

word_counts = {}
for word in words:
  word_counts[word] = word_counts.get(word, 0) + 1

print(word_counts)
```

```
['python', 'for', 'data', 'science', 'is', 'great', 'python', 'is', 'powerful']
{'python': 2, 'for': 1, 'data': 1, 'science': 1, 'is': 2, 'great': 1, 'powerful':
1}
```

In [54]:
```
# Bài 2: Viết chương trình quản lý danh bạ điện thoại bằng dictionary (name → ph
# Khởi tạo
phone_book = {
    "Pi:": "0123456789",
    "hy": "0987654321"

}

# thêm mới
phone_book["piu"] = "0102030405"

# cập nhật
phone_book["pi"] = "0506070809"

# truy cập giá trị
print(phone_book)
print("Số của pi: ", phone_book.get("pi"))
```

```
{'Pi:': '0123456789', 'hy': '0987654321', 'piu': '0102030405', 'pi': '050607080
9'}
Số của pi:  0506070809
```

In [55]:
```python
# Bài 3: Viết chương trình lưu thông tin điểm của nhiều sinh viên vào dictionary
# điểm trung bình.
# Khởi tạo
student_grades = {
    "stA": [10, 9.5, 9],
    "stB": [8, 8.5, 9],
    "stC": [8, 5, 4.5]

}

total_score = 0
total_subjects = 0

for grades in student_grades.values():
  total_score += sum(grades)
  total_subjects += len(grades)

average_score = total_score/total_subjects if total_subjects > 0 else 0

print("điểm trung bình: ", average_score)
```

```
điểm trung bình:  7.944444444444445
```

In [56]:
```python
# Bài 4: Viết chương trình gộp hai dictionary thành một.
dictA = {"a":1, "b":2, "c":3}
dictB = {"d":4, "e":5, "f":6}
merged_dict = dictA | dictB

print(merged_dict)
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5, 'f': 6}
```

In [57]:
```python
# Bài 5: Viết chương trình tìm khóa có giá trị lớn nhất trong dictionary.
my_dict = {
    "Apple": 35,
    "Banana": 20,
    "Orange": 25,
    "Cherry": 40
}

max_key = max(my_dict, key = my_dict.get)
max_value = my_dict[max_key]

print("Khóa có giá trị lớn nhất: ", max_value)
```

```
Khóa có giá trị lớn nhất:  40
```

In [58]:
```python
# Bài 6: Viết chương trình tìm khóa có giá trị nhỏ nhất trong dictionary.
my_dict = {
    "Apple": 35,
    "Banana": 20,
    "Orange": 25,
    "Cherry": 40
}

min_key = min(my_dict, key = my_dict.get)
min_value = my_dict[min_key]
```

```
print("Khóa có giá trị nhỏ nhất: ", min_value)
```

```
Khóa có giá trị nhỏ nhất:  20
```

In [59]:
```
# Bài 7: Viết chương trình đảo ngược dictionary (key thành value, value thành ke
capitals = {
"Maharashtra": "Mumbai",
"Gujarat": "Gandhinagar",
"Telangana": "Hyderabad"
}

dict_dao_nguoc = {value:key for key, value in capitals.items()}
print(dict_dao_nguoc)
```

```
{'Mumbai': 'Maharashtra', 'Gandhinagar': 'Gujarat', 'Hyderabad': 'Telangana'}
```

In [60]:
```
# Bài 8: Viết chương trình lọc ra các cặp khóa-giá trị có giá trị lớn hơn 50 từ
name_score = {
    "Apple": 35,
    "Banana": 90,
    "Orange": 65,
    "Cherry": 50
}

fill_mask = {name: score for name, score in name_score.items() if score > 50}
print(fill_mask)
```

```
{'Banana': 90, 'Orange': 65}
```

In [61]:
```
# Bài 9: Viết chương trình sắp xếp dictionary theo khóa.
student = {"name": "Alice", "age": 21, "major": "CS"}
sorted_by_key = dict(sorted(student.items()))
print(sorted_by_key)
```

```
{'age': 21, 'major': 'CS', 'name': 'Alice'}
```

In [62]:
```
# Bài 10: Viết chương trình sắp xếp dictionary theo giá trị.
marks = {"Savita": 67, "Imitaz": 88, "Laxman": 91}
sorted_by_value = dict(sorted(marks.items(), key = lambda item:item[1], reverse
print("Dictionary sau khi sắp xếp theo giá trị (giảm dần): ", sorted_by_value)
```

```
Dictionary sau khi sắp xếp theo giá trị (giảm dần):  {'Laxman': 91, 'Imitaz': 88,
'Savita': 67}
```

In [63]:
```
# Bài 11: Viết chương trình gộp danh sách sinh viên từ nhiều dictionary lồng nha
# dictionary chung.
dict1 = {"SV001": "Alice", "SV002": "Bob"}
dict2 = {"SV003": "Charlie", "SV004": "John"}
dict3 = {"SV005": "Eve"}

merged_dict = dict1 | dict2 | dict3
print(merged_dict)
```

```
{'SV001': 'Alice', 'SV002': 'Bob', 'SV003': 'Charlie', 'SV004': 'John', 'SV005':
'Eve'}
```

In [64]:
```
# Bài 12: Viết chương trình quản lý sản phẩm trong cửa hàng (tên sản phẩm, giá,
# bằng dictionary.
store = {
    "Laptop" : {"price": 1400, "quantity": 20},
    "Mouse" : {"price": 100, "quantity": 45},
```

```
    "Volume" : {"price": 1200, "quantity": 10}
}

print(f"Tồn kho Laptop: {store['Laptop']['quantity']} chiếc")
```

```
Tồn kho Laptop: 20 chiếc
```

In [65]:
```
# Bài 13: Viết chương trình tính tổng tất cả các giá trị trong dictionary.
marks = {"Savita": 10, "Imitaz": 9, "Laxman": 8}
total_score = sum(marks.values())
print(total_score)
```

```
27
```

In [66]:
```
# Bài 14: Viết chương trình kiểm tra xem một dictionary có rỗng hay không.
dict_check = {"Savita": 10, "Imitaz": 9, "Laxman": 8}
if not dict_check:
  print("Dictionary rỗng")
else:
  print("Dictionary không rỗng")
```

```
Dictionary không rỗng
```

In [67]:
```
# Bài 15: Viết chương trình tạo dictionary từ 2 danh sách: một danh sách khóa và
# sách giá trị.
keys = ["name", "age", "major"]
values = ["Alice", "21", "DS"]
student_info = dict(zip(keys, values))
print(student_info)
```

```
{'name': 'Alice', 'age': '21', 'major': 'DS'}
```

In [68]:
```
# Bài 16: Viết chương trình loại bỏ các phần tử có giá trị trùng lặp trong dicti
data = {"a": 10, "b": 20, "c": 10, "d": 30, "e": 25}
unique_value_dict = {}
seen_values = set()
for key, value in data.items():
  if value not in seen_values:
    unique_value_dict[key] = value
    seen_values.add(value)

print("Sau khi loại bỏ các giá trị trùng lặp: ", unique_value_dict)
```

```
Sau khi loại bỏ các giá trị trùng lặp:  {'a': 10, 'b': 20, 'd': 30, 'e': 25}
```

In [69]:
```
# Bài 17: Viết chương trình chuyển một danh sách tuple thành dictionary.
list_of_tuple = [("Apple", 1), ("Banana", 2), ("Orange", 3)]
new_dict = dict(list_of_tuple)
print(new_dict)
```

```
{'Apple': 1, 'Banana': 2, 'Orange': 3}
```

In [1]:
```
# Bài 18: Viết chương trình chuyển dictionary thành danh sách tuple.
student = {"name": "Alice", "age": 21, "major": "CS"}
list_of_tuple = list(student.items())
print(list_of_tuple)
```

```
[('name', 'Alice'), ('age', 21), ('major', 'CS')]
```

In [2]:
```
# Bài 19: Viết chương trình cập nhật nhiều phần tử trong dictionary cùng lúc bằn
student = {"name": "Alice", "age": 21, "major": "CS"}
student.update({
    "age":23,
```

```
    "email": "232800xx@hcmus.edu.vn",
    "year":"2023"
})

print(student)
```

```
{'name': 'Alice', 'age': 23, 'major': 'CS', 'email': '232800xx@hcmus.edu.vn', 'ye
ar': '2023'}
```

In [3]:
```
# Bài 20: Viết chương trình tìm tất cả các khóa có cùng giá trị trong dictionary
data= {"a": 10, "b": 20, "c":10, "d":30, "e": 25, "f":30}
result = {}
for key, value in data.items():
  result.setdefault(value, []).append(key)

print("Các khóa có cùng giá trị: ", result)
```

```
Các khóa có cùng giá trị:  {10: ['a', 'c'], 20: ['b'], 30: ['d', 'f'], 25: ['e']}
```

Bài số 11: Kiểu dữ liệu Mảng trong python- Python Arrays

In [4]:
```
# Bài 1: Viết chương trình nhập vào một danh sách các số nguyên và in ra danh sá
# khi đã sắp xếp tăng dần.
number = input("Nhập vào danh sách số nguyên cách nhau dấu cách: ")
num_list = []
num_list = [int(x.strip()) for x in number.split(' ') if x.strip()]
num_list.sort()

print("Danh sách đã sắp xếp tăng dần: ", num_list)
```

```
Nhập vào danh sách số nguyên cách nhau dấu cách: 5 4 3 7 23 56 43 12
Danh sách đã sắp xếp tăng dần:  [3, 4, 5, 7, 12, 23, 43, 56]
```

In [5]:
```
# Bài 2: Viết chương trình nhập vào một danh sách các số nguyên và in ra danh sá
numbers = [1, 5, 3, 2, 5, 1, 4, 3, 2, 8]

unique_numbers = list(set(numbers))

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách sau khi loại bỏ trùng lặp: {unique_numbers}")
```

```
Danh sách ban đầu: [1, 5, 3, 2, 5, 1, 4, 3, 2, 8]
Danh sách sau khi loại bỏ trùng lặp: [1, 2, 3, 4, 5, 8]
```

In [7]:
```
from collections import Counter

# Bài 3: Viết chương trình nhập vào một danh sách và tìm giá trị xuất hiện nhiều
data = [4, 1, 5, 2, 4, 4, 3, 2, 1, 4]

counts = Counter(data)

most_common_item = counts.most_common(1)

if most_common_item:
    value, count = most_common_item[0]
    print(f"Danh sách: {data}")
    print(f"Giá trị xuất hiện nhiều nhất: {value} (Số lần: {count})")
```

```
else:
    print("Danh sách rỗng.")
```

```
Danh sách: [4, 1, 5, 2, 4, 4, 3, 2, 1, 4]
Giá trị xuất hiện nhiều nhất: 4 (Số lần: 4)
```

In [8]:
```
# Bài 4: Viết chương trình nhập vào một danh sách và in ra tất cả phần tử duy nh
data = [4, 1, 5, 2, 4, 4, 3, 2, 1, 4, 6]

counts = Counter(data)

unique_only = [item for item, count in counts.items() if count == 1]

print(f"Danh sách: {data}")
print(f"Các phần tử chỉ xuất hiện 1 lần: {unique_only}")
```

```
Danh sách: [4, 1, 5, 2, 4, 4, 3, 2, 1, 4, 6]
Các phần tử chỉ xuất hiện 1 lần: [5, 3, 6]
```

In [9]:
```
# Bài 5: Viết chương trình nối hai mảng số nguyên rồi sắp xếp mảng kết quả.
arr1 = [10, 20, 5, 15]
arr2 = [30, 2, 8, 25]

merged_arr = arr1 + arr2

merged_arr.sort()

print(f"Mảng 1: {arr1}, Mảng 2: {arr2}")
print(f"Mảng sau khi nối và sắp xếp: {merged_arr}")
```

```
Mảng 1: [10, 20, 5, 15], Mảng 2: [30, 2, 8, 25]
Mảng sau khi nối và sắp xếp: [2, 5, 8, 10, 15, 20, 25, 30]
```

In [10]:
```
# Bài 6: Viết chương trình đảo ngược một mảng mà không dùng hàm có sẵn (reverse(
data = [1, 2, 3, 4, 5]
reversed_data = []

for i in range(len(data) - 1, -1, -1):
    reversed_data.append(data[i])

print(f"Danh sách ban đầu: {data}")
print(f"Danh sách đảo ngược thủ công: {reversed_data}")
```

```
Danh sách ban đầu: [1, 2, 3, 4, 5]
Danh sách đảo ngược thủ công: [5, 4, 3, 2, 1]
```

In [11]:
```
# Bài 7: Viết chương trình nhập vào một danh sách và in ra danh sách chỉ gồm số
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

even_numbers = [x for x in numbers if x % 2 == 0]

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách số chẵn: {even_numbers}")
```

```
Danh sách ban đầu: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Danh sách số chẵn: [2, 4, 6, 8, 10]
```

In [12]:
```
# Bài 8: Viết chương trình nhập vào một danh sách và in ra danh sách chỉ gồm số
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

odd_numbers = [x for x in numbers if x % 2 != 0]
```

```
print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách số lẻ: {odd_numbers}")
```

```
Danh sách ban đầu: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Danh sách số lẻ: [1, 3, 5, 7, 9]
```

In [13]:
```
# Bài 9: Viết chương trình xoá tất cả các số âm trong một danh sách số nguyên.
numbers = [10, -5, 20, 0, -15, 30]

non_negative_numbers = [x for x in numbers if x >= 0]

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách sau khi xoá số âm: {non_negative_numbers}")
```

```
Danh sách ban đầu: [10, -5, 20, 0, -15, 30]
Danh sách sau khi xoá số âm: [10, 20, 0, 30]
```

In [14]:
```
# Bài 10: Viết chương trình nhập một danh sách và nhân đôi giá trị tất cả phần t
numbers = [1, 2, 3, 4, 5]

doubled_numbers = [x * 2 for x in numbers]

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách sau khi nhân đôi: {doubled_numbers}")
```

```
Danh sách ban đầu: [1, 2, 3, 4, 5]
Danh sách sau khi nhân đôi: [2, 4, 6, 8, 10]
```

In [15]:
```
# Bài 11: Viết chương trình nhập một danh sách và thay thế tất cả số 0 bằng số 1
numbers = [1, 0, 5, 0, 3, 0, 2]

replaced_numbers = [1 if x == 0 else x for x in numbers]

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách sau khi thay thế 0 bằng 1: {replaced_numbers}")
```

```
Danh sách ban đầu: [1, 0, 5, 0, 3, 0, 2]
Danh sách sau khi thay thế 0 bằng 1: [1, 1, 5, 1, 3, 1, 2]
```

In [16]:
```
import math


# Bài 12: Viết chương trình nhập một danh sách và tính tích các phần tử.
numbers = [1, 2, 3, 4, 5]

product = math.prod(numbers)

print(f"Danh sách: {numbers}")
print(f"Tích các phần tử: {product}")
```

```
Danh sách: [1, 2, 3, 4, 5]
Tích các phần tử: 120
```

In [17]:
```
# Bài 13: Viết chương trình nhập một danh sách và kiểm tra xem nó có đối xứng (p
list1 = [1, 2, 3, 2, 1]
list2 = [1, 2, 3, 4, 5]

is_palindrome1 = list1 == list1[::-1]
is_palindrome2 = list2 == list2[::-1]

print(f"List {list1} là Palindrome: {is_palindrome1}")
print(f"List {list2} là Palindrome: {is_palindrome2}")
```

```
List [1, 2, 3, 2, 1] là Palindrome: True
List [1, 2, 3, 4, 5] là Palindrome: False
```

In [18]:
```python
# Bài 14: Viết chương trình nhập một danh sách và tìm 2 phần tử có tổng lớn nhất
numbers = [10, 5, 20, 8, 15]

numbers.sort(reverse=True)

max1 = numbers[0]
max2 = numbers[1]

print(f"Danh sách đã sắp xếp: {numbers}")
print(f"Hai phần tử có tổng lớn nhất: {max1} và {max2} (Tổng: {max1 + max2})")
```

```
Danh sách đã sắp xếp: [20, 15, 10, 8, 5]
Hai phần tử có tổng lớn nhất: 20 và 15 (Tổng: 35)
```

In [19]:
```python
# Bài 15: Viết chương trình nhập một danh sách và tìm 2 phần tử có tổng nhỏ nhất
numbers = [10, 5, 20, 8, 15]
numbers.sort()

min1 = numbers[0]
min2 = numbers[1]

print(f"Danh sách đã sắp xếp: {numbers}")
print(f"Hai phần tử có tổng nhỏ nhất: {min1} và {min2} (Tổng: {min1 + min2})")
```

```
Danh sách đã sắp xếp: [5, 8, 10, 15, 20]
Hai phần tử có tổng nhỏ nhất: 5 và 8 (Tổng: 13)
```

In [20]:
```python
# Bài 16: Viết chương trình nhập vào một danh sách chuỗi và sắp xếp theo độ dài
strings = ["data", "science", "AI", "engineer", "python"]

sorted_by_length = sorted(strings, key=len)

print(f"Danh sách ban đầu: {strings}")
print(f"Danh sách sắp xếp theo độ dài: {sorted_by_length}")
```

```
Danh sách ban đầu: ['data', 'science', 'AI', 'engineer', 'python']
Danh sách sắp xếp theo độ dài: ['AI', 'data', 'python', 'science', 'engineer']
```

In [21]:
```python
# Bài 17: Viết chương trình nhập vào một danh sách chuỗi và lọc ra các chuỗi bắt
strings = ["apple", "Banana", "cat", "algorithm", "Array", "zebra"]

filtered_strings = [
    s for s in strings
    if s.lower().startswith('a')
]

print(f"Danh sách ban đầu: {strings}")
print(f"Chuỗi bắt đầu bằng 'a' (không phân biệt): {filtered_strings}")
```

```
Danh sách ban đầu: ['apple', 'Banana', 'cat', 'algorithm', 'Array', 'zebra']
Chuỗi bắt đầu bằng 'a' (không phân biệt): ['apple', 'algorithm', 'Array']
```

In [22]:
```python
# Bài 18: Viết chương trình nhập vào một danh sách số nguyên và xoá tất cả phần
data = [1, 5, 3, 2, 5, 1, 4, 3, 2, 8]
result = []
seen = set()

for item in data:
    if item not in seen:
```

```
        seen.add(item)
        result.append(item)

print(f"Danh sách ban đầu: {data}")
print(f"Xoá trùng lặp, giữ nguyên thứ tự: {result}")
```

```
Danh sách ban đầu: [1, 5, 3, 2, 5, 1, 4, 3, 2, 8]
Xoá trùng lặp, giữ nguyên thứ tự: [1, 5, 3, 2, 4, 8]
```

In [23]:
```
# Bài 19: Viết chương trình nhập vào một danh sách số nguyên và in ra danh sách

def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

numbers = [2, 7, 10, 11, 15, 17, 20]

prime_numbers = [x for x in numbers if is_prime(x)]

print(f"Danh sách ban đầu: {numbers}")
print(f"Danh sách số nguyên tố: {prime_numbers}")
```

```
Danh sách ban đầu: [2, 7, 10, 11, 15, 17, 20]
Danh sách số nguyên tố: [2, 7, 11, 17]
```

In [24]:
```
# Bài 20: Viết chương trình nhập vào một danh sách số nguyên và in ra phần tử lớ
numbers = [10, 5, 20, 8, 15, 20]

unique_numbers = list(set(numbers))

unique_numbers.sort()

if len(unique_numbers) < 2:
    print("Không đủ phần tử để tìm phần tử lớn thứ hai.")
else:
    second_largest = unique_numbers[-2]
    print(f"Danh sách ban đầu: {numbers}")
    print(f"Danh sách không trùng lặp: {unique_numbers}")
    print(f"Phần tử lớn thứ hai: {second_largest}")
```

```
Danh sách ban đầu: [10, 5, 20, 8, 15, 20]
Danh sách không trùng lặp: [5, 8, 10, 15, 20]
Phần tử lớn thứ hai: 15
```