Nguyễn Hồng Yến - 23280099

Bài số 6: Xử lý Chuỗi ký tự (strings) trong Python

Phần A: Bài tập Nhận biết và Ghi nhớ (30 câu)

In [ ]:
```
# Bài 1: Khai báo một chuỗi "Hello Python" và in ra màn hình.
string = "Hello Python"
print(string)
```

```
Hello Python
```

In [ ]:
```
# Bài 2: In ký tự đầu tiên của chuỗi "OpenAI".
string = "OpenAI"
first_character = string[0]
print(first_character)
```

```
O
```

In [ ]:
```
# Bài 3: In ký tự cuối cùng của chuỗi "ChatGPT".
string = "ChatGPT"
last_charactor = string[-1]
print(last_charactor)
```

```
T
```

In [ ]:
```
# Bài 4: Lấy 5 ký tự đầu tiên của chuỗi "Data Science".
string = "Data Science"
string[:5]
```

Out[ ]:
```
'Data '
```

In [ ]:
```
# Bài 5: Lấy 4 ký tự cuối của chuỗi "Machine Learning".
string = "Machine Learning"
string[-4:]
```

Out[ ]:
```
'ning'
```

In [ ]:
```
# Bài 6: In chiều dài của chuỗi "Artificial Intelligence".
string = "Artificial Intelligence"
string_length = len(string)
print(string_length)
```

```
23
```

In [ ]:
```
# Bài 7: Chuyển chuỗi "python programming" thành chữ in hoa.
string = "python programming"
string.upper()
```

Out[ ]:
```
'PYTHON PROGRAMMING'
```

In [ ]:
```
# Bài 8: Chuyển chuỗi "PYTHON PROGRAMMING" thành chữ thường.
string = "PYTHON PROGRAMMING"
string.lower()
```

Out[ ]: 'python programming'

In [ ]:
```
# Bài 9: Viết chương trình đổi ký tự đầu của chuỗi "hello" thành chữ in hoa.
def ham(string):
  string = string.capitalize()
  return string

ham("hello")
```

Out[ ]: 'Hello'

In [ ]:
```
# Bài 10: In chuỗi "banana" với mỗi ký tự trên một dòng.
string = "banana"
for char in string:
  print(char)
```

```
b
a
n
a
n
a
```

In [ ]:
```
# Bài 11: Ghép hai chuỗi "Data" và "Science" bằng toán tử +.
str1 = "Data"
str2= "Science"
str3 = str1 + str2
print(str3)
```

```
DataScience
```

In [ ]:
```
# Bài 12: Ghép hai chuỗi "Deep" và "Learning" bằng f-string.
str1 = "Deep"
str2 = "Learning"
str3 = f'str:  {str1+str2}'
print(str3)
```

```
str:  DeepLearning
```

In [ ]:
```
# Bài 13: Ghép chuỗi "Age:" với số 25 bằng phương thức format().
str = "Age {}"
num = 25
str.format(num)
```

Out[ ]: 'Age 25'

In [ ]:
```
# Bài 14: In chuỗi có chứa dấu nháy kép: "He said, ``Python is fun!'' ".
print(""" "He said, ''Python is fun!''" """)
```

```
"He said, ''Python is fun!''"
```

In [ ]:
```
# Bài 15: In chuỗi có chứa dấu xuống dòng bằng ký tự thoát
string = "Hello\nWorld"
print(string)
```

```
Hello
World
```

In [ ]:
```python
# Bài 16: In chuỗi có chứa ký tự tab bằng ký tự thoát.
string = "My\tname\tis\tPi"
print(string)
```

```
My	name	is	Pi
```

In [ ]:
```python
# Bài 17: Kiểm tra xem chuỗi "apple" có chứa ký tự "a" không.
str = "apple"

def check(string, char):
  if(char in string):
    return True
  else:
    return False

check(str, "a")
```

Out[ ]: True

In [ ]:
```python
# Bài 18: Đếm số lần ký tự "e" xuất hiện trong chuỗi "engineering".
string = "engineering"
freq = string.count("e")

print(freq)
```

```
3
```

In [ ]:
```python
# Bài 19: Tìm vị trí đầu tiên của chuỗi con "data" trong "big data analysis".
str = "big data analysis"
sub_str= "data"

str.find(sub_str)
```

Out[ ]: 4

In [ ]:
```python
# Bài 20: Thay thế "blue" bằng "red" trong chuỗi "The sky is blue".
str = "The sky is blue"
new_str = str.replace("blue", "red")

print(new_str)
```

```
The sky is red
```

In [ ]:
```python
# Bài 21: Loại bỏ khoảng trắng ở đầu và cuối chuỗi "hello ".
str = " hello "
str.strip()
```

Out[ ]: 'hello'

- Bài 22: Tách chuỗi "Python,Java,C++" thành danh sách.
- Bài 23: Nối danh sách ["Python","Java","C++"] thành chuỗi, ngăn cách bởi dấu phẩy.
- Bài 24: Kiểm tra chuỗi "12345" có phải toàn số không.
- Bài 25: Kiểm tra chuỗi "Hello" có phải toàn chữ cái không.
- Bài 26: Viết chương trình đảo ngược chuỗi "Python".
- Bài 27: In ký tự thứ 2 đến thứ 5 của chuỗi "Statistics".

- Bài 28: In chuỗi "Python" 3 lần liên tiếp bằng toán tử *.
- Bài 29: Kiểm tra chuỗi "hello world" có bắt đầu bằng "hello" không.
- Bài 30: Kiểm tra chuỗi "hello world" có kết thúc bằng "world" không.

In [ ]:
```python
# Bài 22: Tách chuỗi "Python,Java,C++" thành danh sách.
str = "Python,Java,C++"
print(str.split(","))


#Bài 23: Nối danh sách ["Python","Java","C++"] thành chuỗi, ngăn cách bởi dấu ph
list = ["Python","Java","C++"]
print(",".join(list))

# Bài 24: Kiểm tra chuỗi "12345" có phải toàn số không.
str = "12345"
print(str.isnumeric())

# Bài 25: Kiểm tra chuỗi "Hello" có phải toàn chữ cái không.
str = "Hello"
print(str.isalpha())

# Bài 26: Viết chương trình đảo ngược chuỗi "Python".
str = "Python"
print(str[::-1])

# Bài 27: In ký tự thứ 2 đến thứ 5 của chuỗi "Statistics".
str = "Statistics"
print(str[1:5])

# Bài 28: In chuỗi "Python" 3 lần liên tiếp bằng toán tử *.
str = "Python"
print(str*3)

# Bài 29: Kiểm tra chuỗi "hello world" có bắt đầu bằng "hello" không.
str = "hello world"
str.startswith("hello")

# Bài 30: Kiểm tra chuỗi "hello world" có kết thúc bằng "world" không.
str = "hello world"
str.endswith("hello")
```

```
['Python', 'Java', 'C++']
Python,Java,C++
True
True
nohtyP
tati
PythonPythonPython
```

Out[ ]: False

Phần B: Bài tập Vận dụng (20 câu)

In [ ]:
```python
# Bài 31: Viết chương trình nhập vào một chuỗi và in ra chuỗi đảo ngược.
def in_chuoi_dao_nguoc(string):
  string = string[::-1]
  return string
```

```
in_chuoi_dao_nguoc("Daylapi")
```

Out[ ]: 'ipalyaD'

In [ ]:
```
# Bài 32: Nhập một chuỗi từ bàn phím và đếm số nguyên âm (a, e, i, o, u).
def dem(string):
  string = string.lower()
  nguyen_am = "aeiou"
  count = 0
  for i in string:
    if i in nguyen_am:
      count = count + 1
  return count

dem("orange")
```

Out[ ]: 3

In [ ]:
```
# Bài 33: Nhập vào họ và tên, in ra tên viết hoa toàn bộ.
full_name = input("Họ tên: ")
fn_upper = full_name.upper()
print(fn_upper)
```

```
Họ tên: nguyen hong yen
NGUYEN HONG YEN
```

In [ ]:
```
# Bài 34: Viết chương trình định dạng chuỗi để in bảng cột: Tên, Tuổi, Nghề nghi
```

In [ ]:
```
# Bài 34: Viết chương trình định dạng chuỗi để in bảng cột: Tên, Tuổi, Nghề nghi

data = [
    ("Pi", 21, "AIE"),
    ("Ha", 21, "DA")
]
print(f"{'Tên': <10} {'Tuổi':<10} {'Nghề nghiệp':<20}")
print("-" * 35)
for name, age, job in data:
  print(f"{name: <10} {age:<10} {job:20}")
```

```
Tên        Tuổi       Nghề nghiệp
-----------------------------------
Pi         21         AIE
Ha         21         DA
```

In [ ]:
```

```

In [ ]:
```
# Bài 35: Viết chương trình nhập chuỗi và loại bỏ tất cả khoảng trắng.
def loai_bo_khoang_trang(str):
  input("Nhập chuỗi: ")
  str_new = str.replace(' ', '')
  return str_new

loai_bo_khoang_trang("hong yen")
```

```
Nhập chuỗi: hong yen
```

Out[ ]: 'hongyen'

In [ ]:
```
# Bài 36: Viết chương trình thay thế tất cả nguyên âm trong chuỗi bằng dấu "*".
def thay_the_nguyen_am(str1):
  nguyen_am = "uioae"
  replacement_char = "*"
  # str_new = ""
  for i in nguyen_am:
    str1 = str1.replace(i, replacement_char)
  return str1

thay_the_nguyen_am("orange")
```

Out[ ]:
```
'*r*ng*'
```

In [ ]:
```
# Bài 37: Viết chương trình nhập chuỗi và đếm số lượng từ
input_str = input("Nhập chuỗi: ")
words = input_str.split()
word_count = len(words)
print(words)
```

```
Nhập chuỗi: day la con cho pi !!
['day', 'la', 'con', 'cho', 'pi', '!!']
```

In [ ]:
```
# Bài 38: Viết chương trình nhập chuỗi và in từ dài nhất.
def tu_dai_nhat(s):
  words = s.split()
  word_count = 0
  max_length = 0
  res = words[0]
  for word in words:
    word_count += 1
    if len(word) > max_length:
      max_length = len(word)
      res = word
  return res

tu_dai_nhat("nguyen hong yen")
```

Out[ ]:
```
'nguyen'
```

In [ ]:
```
# Bài 39: Viết chương trình kiểm tra xem chuỗi nhập vào có phải là palindrome (đ
# không.
def is_palindrome(s):
  s1 = s.lower()
  s_check = s1[::-1]
  if (s1 == s_check):
    return True
  return False

print(is_palindrome("ppipp"))
print(is_palindrome("hongyen"))
```

```
True
False
```

In [ ]:
```
# Bài 40: Viết chương trình nhập email và kiểm tra có chứa ký tự "@" hay không.
def check(email):
  for i in email:
    if i == "@":
      return True
  return False
```

```
check("Xin chao, toi den tu JobsGo")
```

Out[ ]: False

In [ ]:
```
# Bài 41: Viết chương trình nhập số điện thoại và kiểm tra có đúng 10 chữ số khô
def check_phone_number(num):
  if(not num.isnumeric()):
    return False
  else:
     num_length = len(num)
     if (num_length == 10):
      return True
     return False

print(check_phone_number("0387010307"))
print(check_phone_number("pi"))
print(check_phone_number("0387010307977"))
```

```
True
False
False
```

In [2]:
```
# Bài 42: Viết chương trình nhập mật khẩu, kiểm tra có chứa ít nhất một chữ hoa,
# thường, một số và một ký tự đặc biệt.
password = input("Nhập mật khẩu: ")

has_upper = False
has_lower = False
has_digit = False
has_special = False

special_chars = "!@#$%^&*()-_=+[{]}\|;:'\",<.>/?`~"

for ch in password:
    if ch.isupper():
        has_upper = True
    elif ch.islower():
        has_lower = True
    elif ch.isdigit():
        has_digit = True
    elif ch in special_chars:
        has_special = True

if has_upper and has_lower and has_digit and has_special:
    print("Mật khẩu hợp lệ")
else:
    print("Mật khẩu chưa hợp lệ.")
    print("Yêu cầu: ít nhất 1 chữ hoa, 1 chữ thường, 1 số và 1 ký tự đặc biệt.")
```

```
<>:10: SyntaxWarning: invalid escape sequence '\|'
<>:10: SyntaxWarning: invalid escape sequence '\|'
/tmp/ipython-input-117786476.py:10: SyntaxWarning: invalid escape sequence '\|'
  special_chars = "!@#$%^&*()-_=+[{]}\|;:'\",<.>/?`~"
Nhập mật khẩu: 14022005@
Mật khẩu chưa hợp lệ.
Yêu cầu: ít nhất 1 chữ hoa, 1 chữ thường, 1 số và 1 ký tự đặc biệt.
```

In [3]:
```
# Bài 43: Viết chương trình in ra tất cả các ký tự khác nhau trong một chuỗi.
def unique_chars(string):
```

```
  return set(string)

print(unique_chars("programming"))
```

```
{'r', 'm', 'g', 'a', 'n', 'o', 'i', 'p'}
```

In [4]:
```
# Bài 44: Viết chương trình chuẩn hóa tên (viết hoa chữ cái đầu mỗi từ).
def normalize_name(name):
  return name.title()

print(normalize_name("nguyen hong yen"))
```

```
Nguyen Hong Yen
```

In [5]:
```
# Bài 45: Viết chương trình nhập chuỗi và in ra chuỗi sau khi bỏ các ký tự trùng
def remove_duplicates(string):
  seen = set()
  result = []
  for char in string:
    if char not in seen:
      seen.add(char)
      result.append(char)
  return "".join(result)

print(remove_duplicates("programming"))
```

```
progamin
```

In [6]:
```
# Bài 46: Viết chương trình in ra tất cả các chuỗi con (substring) của một chuỗi
def print_substrings(string):
  n = len(string)
  for i in range(n):
    for j in range(i, n):
      print(string[i:j+1])

print_substrings("abc")
```

```
a
ab
abc
b
bc
c
```

In [7]:
```
# Bài 47: Viết chương trình mã hóa chuỗi theo quy tắc thay mỗi ký tự bằng ký tự
# trong bảng ASCII.
def encode_string(string):
  encoded_list = [chr(ord(char) + 1) for char in string]
  return "".join(encoded_list)

print(encode_string("abc"))
```

```
bcd
```

In [8]:
```
# Bài 48: Viết chương trình giải mã chuỗi đã mã hóa theo quy tắc trên.
def decode_string(string):
  decoded_list = [chr(ord(char) - 1) for char in string]
  return "".join(decoded_list)

print(decode_string("bcd"))
```

```
abc
```

In [9]:
```
# Bài 49: Viết chương trình nhập danh sách các chuỗi và sắp xếp theo thứ tự bảng
def sort_strings(string_list):
  string_list.sort()
  return string_list

my_list = ["banana", "apple", "cherry"]
print(sort_strings(my_list))
```

```
['apple', 'banana', 'cherry']
```

In [10]:
```
# Bài 50: Viết chương trình nhập chuỗi và đếm tần suất xuất hiện của từng ký tự.
def count_char_frequency(string):
  frequency = {}
  for char in string:
    if char in frequency:
      frequency[char] += 1
    else:
      frequency[char] = 1
  return frequency

print(count_char_frequency("programming"))
```

```
{'p': 1, 'r': 2, 'o': 1, 'g': 2, 'a': 1, 'm': 2, 'i': 1, 'n': 1}
```

Bài số 7: Danh sách trong Python - Python Lists

Phần A. Bài tập Nhận biết và Ghi nhớ (30 câu)

In [ ]:
```

```

In [11]:
```
# Bài 1: Tạo một list fruits gồm: ["apple", "banana", "cherry"] và in ra màn hìn
fruits = ["apple", "banana", "cherry"]
print(fruits)
```

```
['apple', 'banana', 'cherry']
```

In [12]:
```
# Bài 2: In ra phần tử đầu tiên của list fruits.
fruits = ["apple", "banana", "cherry"]
print(fruits[0])
```

```
apple
```

In [13]:
```
# Bài 3: In ra phần tử cuối cùng của list fruits.
fruits = ["apple", "banana", "cherry"]
print(fruits[-1])
```

```
cherry
```

In [14]:
```
# Bài 4: Dùng chỉ số âm để in phần tử "banana" trong list.
fruits = ["apple", "banana", "cherry"]
print(fruits[-2])
```

```
banana
```

In [15]:
```
# Bài 5: Thay đổi phần tử thứ hai của list fruits thành "mango".
fruits = ["apple", "banana", "cherry"]
```

```
fruits[1] = "mango"
print(fruits)
```

```
['apple', 'mango', 'cherry']
```

In [16]:
```
# Bài 6: Thêm phần tử "orange" vào cuối list bằng append().
fruits = ["apple", "banana", "cherry"]
fruits.append("orange")
print(fruits)
```

```
['apple', 'banana', 'cherry', 'orange']
```

In [17]:
```
# Bài 7: Thêm phần tử "kiwi" vào vị trí thứ hai bằng insert().
fruits = ["apple", "banana", "cherry"]
fruits.insert(1, "kiwi")
print(fruits)
```

```
['apple', 'kiwi', 'banana', 'cherry']
```

In [18]:
```
# Bài 8: Xóa phần tử cuối cùng của list bằng pop().
fruits = ["apple", "banana", "cherry"]
fruits.pop()
print(fruits)
```

```
['apple', 'banana']
```

In [19]:
```
# Bài 9: Xóa phần tử "apple" bằng remove().
fruits = ["apple", "banana", "cherry"]
fruits.remove("apple")
print(fruits)
```

```
['banana', 'cherry']
```

In [20]:
```
# Bài 10: Tạo một list số nguyên từ 1 đến 5 và in ra.
numbers = [1, 2, 3, 4, 5]
print(numbers)
```

```
[1, 2, 3, 4, 5]
```

In [ ]:
```

```

In [21]:
```
# Bài 11: Duyệt list số nguyên bằng vòng lặp for và in từng phần tử.
numbers = [1, 2, 3, 4, 5]
for number in numbers:
  print(number)
```

```
1
2
3
4
5
```

In [22]:
```
# Bài 12: Duyệt list số nguyên bằng vòng lặp while.
numbers = [1, 2, 3, 4, 5]
i = 0
while i < len(numbers):
  print(numbers[i])
  i += 1
```

```
1
2
3
4
5
```

In [23]:
```python
# Bài 13: Sử dụng cú pháp for ... in range() để in chỉ số và phần tử.
fruits = ["apple", "banana", "cherry"]
for i in range(len(fruits)):
  print(f"Index: {i}, Element: {fruits[i]}")
```

```
Index: 0, Element: apple
Index: 1, Element: banana
Index: 2, Element: cherry
```

In [24]:
```python
# Bài 14: Sử dụng list comprehension để tạo list bình phương các số từ 1 đến 5.
squares = [x**2 for x in range(1, 6)]
print(squares)
```

```
[1, 4, 9, 16, 25]
```

In [25]:
```python
# Bài 15: Sử dụng list comprehension để tạo list gồm các số chẵn từ 1 đến 20.
evens = [x for x in range(1, 21) if x % 2 == 0]
print(evens)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

In [26]:
```python
# Bài 16: Sắp xếp list số nguyên theo thứ tự tăng dần bằng sort().
numbers = [5, 2, 8, 1, 9]
numbers.sort()
print(numbers)
```

```
[1, 2, 5, 8, 9]
```

In [27]:
```python
# Bài 17: Sắp xếp list số nguyên theo thứ tự giảm dần.
numbers = [5, 2, 8, 1, 9]
numbers.sort(reverse=True)
print(numbers)
```

```
[9, 8, 5, 2, 1]
```

In [28]:
```python
# Bài 18: Tạo một bản sao của list số nguyên bằng copy().
original_list = [1, 2, 3]
copied_list = original_list.copy()
print(copied_list)
```

```
[1, 2, 3]
```

In [29]:
```python
# Bài 19: Tạo một bản sao của list số nguyên bằng list().
original_list = [1, 2, 3]
copied_list = list(original_list)
print(copied_list)
```

```
[1, 2, 3]
```

In [30]:
```python
# Bài 20: Nối hai list [1,2,3] và [4,5,6] lại với nhau bằng phép cộng.
list1 = [1, 2, 3]
list2 = [4, 5, 6]
combined_list = list1 + list2
print(combined_list)
```

```
[1, 2, 3, 4, 5, 6]
```

In [31]:
```
# Bài 21: Nối hai list bằng extend().
list1 = [1, 2, 3]
list2 = [4, 5, 6]
list1.extend(list2)
print(list1)
```

```
[1, 2, 3, 4, 5, 6]
```

In [32]:
```
# Bài 22: Đếm số lần xuất hiện của phần tử "banana" trong list.
fruits = ["apple", "banana", "cherry", "banana"]
count = fruits.count("banana")
print(count)
```

```
2
```

In [33]:
```
# Bài 23: Tìm chỉ số đầu tiên của phần tử "cherry" trong list.
fruits = ["apple", "banana", "cherry", "banana"]
index = fruits.index("cherry")
print(index)
```

```
2
```

In [34]:
```
# Bài 24: Kiểm tra xem "apple" có trong list không, kết quả trả về True/False.
fruits = ["apple", "banana", "cherry"]
print("apple" in fruits)
```

```
True
```

In [35]:
```
# Bài 25: Sử dụng clear() để xóa tất cả phần tử trong list.
fruits = ["apple", "banana", "cherry"]
fruits.clear()
print(fruits)
```

```
[]
```

In [36]:
```
# Bài 26: Tạo một list chứa 5 chuỗi bất kỳ và in độ dài từng chuỗi.
string_list = ["python", "java", "c++", "javascript", "html"]
for s in string_list:
  print(len(s))
```

```
6
4
3
10
4
```

In [37]:
```
# Bài 27: Tạo một list chứa số thực và in ra tổng các phần tử.
float_list = [1.1, 2.2, 3.3, 4.4]
total = sum(float_list)
print(total)
```

```
11.0
```

In [38]:
```
# Bài 28: Tạo một list số nguyên bất kỳ, in ra giá trị lớn nhất và nhỏ nhất.
numbers = [10, 5, 20, 2, 15]
print(f"Max: {max(numbers)}")
print(f"Min: {min(numbers)}")
```

```
Max: 20
Min: 2
```

In [39]:
```
# Bài 29: Tạo một list số nguyên bất kỳ, in ra độ dài list bằng len().
numbers = [10, 5, 20, 2, 15]
```

```
print(len(numbers))
```

```
5
```

In [40]:
```
# Bài 30: Sử dụng reverse() để đảo ngược thứ tự các phần tử trong list.
numbers = [1, 2, 3, 4, 5]
numbers.reverse()
print(numbers)
```

```
[5, 4, 3, 2, 1]
```

Phần B. Bài tập Vận dụng (20 câu)

In [ ]:
```

```

In [41]:
```
# Bài 1: Viết chương trình nhập vào 5 số nguyên từ bàn phím, lưu vào list và in
numbers = []
for i in range(5):
    num = int(input(f"Nhập số nguyên thứ {i+1}: "))
    numbers.append(num)
print(numbers)
```

```
Nhập số nguyên thứ 1: 5
Nhập số nguyên thứ 2: 6
Nhập số nguyên thứ 3: 67
Nhập số nguyên thứ 4: 34
Nhập số nguyên thứ 5: 1
[5, 6, 67, 34, 1]
```

In [42]:
```
# Bài 2: Viết chương trình nhập vào một list số nguyên, tính và in ra trung bình
numbers = [10, 20, 30, 40, 50]
average = sum(numbers) / len(numbers)
print(f"Trung bình cộng: {average}")
```

```
Trung bình cộng: 30.0
```

In [43]:
```
# Bài 3: Viết chương trình nhập vào một list số nguyên, in ra các số lẻ.
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
odd_numbers = [num for num in numbers if num % 2 != 0]
print(f"Các số lẻ: {odd_numbers}")
```

```
Các số lẻ: [1, 3, 5, 7, 9]
```

In [44]:
```
# Bài 4: Viết chương trình nhập vào một list số nguyên, in ra các số nguyên tố.
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10]
prime_numbers = [num for num in numbers if is_prime(num)]
print(f"Các số nguyên tố: {prime_numbers}")
```

```
Các số nguyên tố: [2, 3, 5, 7]
```

In [45]:
```
# Bài 5: Viết chương trình nhập vào một list số nguyên, in ra các số lớn hơn giá
numbers = [10, 20, 30, 40, 50]
average = sum(numbers) / len(numbers)
```

```
greater_than_average = [num for num in numbers if num > average]
print(f"Các số lớn hơn trung bình ({average}): {greater_than_average}")
```

```
Các số lớn hơn trung bình (30.0): [40, 50]
```

In [46]:
```
# Bài 6: Viết chương trình nhập một list chuỗi, in ra chuỗi dài nhất.
string_list = ["apple", "banana", "cherry", "date"]
longest_string = max(string_list, key=len)
print(f"Chuỗi dài nhất: {longest_string}")
```

```
Chuỗi dài nhất: banana
```

In [47]:
```
# Bài 7: Viết chương trình nhập một list chuỗi, in ra chuỗi có nhiều ký tự "a" n
def count_a(string):
    return string.lower().count('a')

string_list = ["apple", "banana", "cherry", "date"]
string_with_most_a = max(string_list, key=count_a)
print(f"Chuỗi có nhiều ký tự 'a' nhất: {string_with_most_a}")
```

```
Chuỗi có nhiều ký tự 'a' nhất: banana
```

In [48]:
```
# Bài 8: Viết chương trình xóa tất cả phần tử trùng lặp trong một list.
numbers = [1, 2, 2, 3, 4, 4, 5]
unique_numbers = list(set(numbers))
print(f"List sau khi xóa phần tử trùng lặp: {unique_numbers}")
```

```
List sau khi xóa phần tử trùng lặp: [1, 2, 3, 4, 5]
```

In [49]:
```
# Bài 9: Viết chương trình trộn ngẫu nhiên các phần tử trong một list (gợi ý: dù
import random

numbers = [1, 2, 3, 4, 5]
random.shuffle(numbers)
print(f"List sau khi trộn ngẫu nhiên: {numbers}")
```

```
List sau khi trộn ngẫu nhiên: [2, 4, 5, 3, 1]
```

In [50]:
```
# Bài 10: Viết chương trình sắp xếp một list chuỗi theo thứ tự từ điển.
string_list = ["banana", "apple", "cherry"]
string_list.sort()
print(f"List chuỗi sau khi sắp xếp: {string_list}")
```

```
List chuỗi sau khi sắp xếp: ['apple', 'banana', 'cherry']
```

In [51]:
```
# Bài 11: Viết chương trình tạo list số nguyên từ 1 đến 100, sau đó lọc ra các s
numbers = [x for x in range(1, 101) if x % 7 == 0]
print(f"Các số chia hết cho 7: {numbers}")
```

```
Các số chia hết cho 7: [7, 14, 21, 28, 35, 42, 49, 56, 63, 70, 77, 84, 91, 98]
```

In [52]:
```
# Bài 12: Viết chương trình nối hai list số nguyên, sau đó sắp xếp tăng dần.
list1 = [5, 2, 8]
list2 = [1, 9, 4]
combined_list = list1 + list2
combined_list.sort()
print(f"List sau khi nối và sắp xếp: {combined_list}")
```

```
List sau khi nối và sắp xếp: [1, 2, 4, 5, 8, 9]
```

In [53]:
```
# Bài 13: Viết chương trình nhập hai list chuỗi, in ra list các phần tử chung.
list1 = ["apple", "banana", "cherry"]
list2 = ["banana", "date", "cherry"]
```

```
common_elements = list(set(list1) & set(list2))
print(f"Các phần tử chung: {common_elements}")
```

```
Các phần tử chung: ['cherry', 'banana']
```

In [54]:
```
# Bài 14: Viết chương trình nhập list số nguyên, in ra list bình phương của chún
numbers = [1, 2, 3, 4, 5]
squares = [num**2 for num in numbers]
print(f"List bình phương: {squares}")
```

```
List bình phương: [1, 4, 9, 16, 25]
```

In [55]:
```
# Bài 15: Viết chương trình nhập list số nguyên, in ra list chứa giá trị tuyệt đ
numbers = [1, -2, 3, -4, 5]
absolute_values = [abs(num) for num in numbers]
print(f"List giá trị tuyệt đối: {absolute_values}")
```

```
List giá trị tuyệt đối: [1, 2, 3, 4, 5]
```

In [56]:
```
# Bài 16: Viết chương trình nhập list số nguyên, kiểm tra xem có số âm không.
numbers = [1, 2, 3, -4, 5]
has_negative = any(num < 0 for num in numbers)
print(f"Có số âm trong list không? {has_negative}")
```

```
Có số âm trong list không? True
```

In [57]:
```
# Bài 17: Viết chương trình nhập list số nguyên, in ra list các phần tử duy nhất
numbers = [1, 2, 2, 3, 4, 4, 5]
unique_numbers = list(dict.fromkeys(numbers))
print(f"List các phần tử duy nhất: {unique_numbers}")
```

```
List các phần tử duy nhất: [1, 2, 3, 4, 5]
```

In [58]:
```
# Bài 18: Viết chương trình nhập list chuỗi, in ra list gồm độ dài của từng chuỗ
string_list = ["apple", "banana", "cherry"]
lengths = [len(s) for s in string_list]
print(f"List độ dài chuỗi: {lengths}")
```

```
List độ dài chuỗi: [5, 6, 6]
```

In [59]:
```
# Bài 19: Viết chương trình nhập list số nguyên, chia list thành 2 list: số chẵn
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_numbers = [num for num in numbers if num % 2 == 0]
odd_numbers = [num for num in numbers if num % 2 != 0]
print(f"List số chẵn: {even_numbers}")
print(f"List số lẻ: {odd_numbers}")
```

```
List số chẵn: [2, 4, 6, 8, 10]
List số lẻ: [1, 3, 5, 7, 9]
```

In [60]:
```
# Bài 20: Viết chương trình nhập list số nguyên, in ra list các phần tử theo thứ
numbers = [1, 2, 3, 4, 5]
reversed_numbers = numbers[::-1]
print(f"List đảo ngược: {reversed_numbers}")
```

```
List đảo ngược: [5, 4, 3, 2, 1]
```

Bài số 8: Dữ liệu tuples trong python - Python tuples

Phần A: Nhận biết và Ghi nhớ (30 câu)

In [61]:
```
# Bài 1: Tạo một tuple chứa các số nguyên từ 1 đến 5.
my_tuple = (1, 2, 3, 4, 5)
print(my_tuple)
```

```
(1, 2, 3, 4, 5)
```

In [62]:
```
# Bài 2: Tạo một tuple chứa các tên môn học: “Toán”, “Lý”, “Hóa”, “Sinh”.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(subjects)
```

```
('Toán', 'Lý', 'Hóa', 'Sinh')
```

In [63]:
```
# Bài 3: In ra tuple đã tạo ở Câu 2.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(subjects)
```

```
('Toán', 'Lý', 'Hóa', 'Sinh')
```

In [64]:
```
# Bài 4: Lấy phần tử đầu tiên của tuple ở Câu 2.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(subjects[0])
```

```
Toán
```

In [65]:
```
# Bài 5: Lấy phần tử cuối cùng của tuple ở Câu 2.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(subjects[-1])
```

```
Sinh
```

In [66]:
```
# Bài 6: Lấy các phần tử từ vị trí 1 đến 3 trong tuple ở Câu 2.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(subjects[1:4])
```

```
('Lý', 'Hóa', 'Sinh')
```

In [67]:
```
# Bài 7: Tạo một tuple có 1 phần tử duy nhất là số 100.
single_element_tuple = (100,)
print(single_element_tuple)
```

```
(100,)
```

In [68]:
```
# Bài 8: Kiểm tra kiểu dữ liệu của tuple ở Câu 7 bằng type().
single_element_tuple = (100,)
print(type(single_element_tuple))
```

```
<class 'tuple'>
```

In [69]:
```
# Bài 9: Dùng vòng lặp for in ra tất cả phần tử trong tuple ở Câu 2.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
for subject in subjects:
  print(subject)
```

```
Toán
Lý
Hóa
Sinh
```

In [70]:
```
# Bài 10: Kiểm tra xem giá trị “Hóa” có trong tuple ở Câu 2 hay không.
subjects = ("Toán", "Lý", "Hóa", "Sinh")
```

```
print("Hóa" in subjects)
```

```
True
```

In [72]:
```
# Bài 11: Tính số phần tử trong tuple ở Câu 2 bằng len().
subjects = ("Toán", "Lý", "Hóa", "Sinh")
print(len(subjects))
```

```
4
```

In [73]:
```
# Bài 12: Tạo 2 tuple: (1,2,3) và (4,5,6), sau đó nối chúng lại.
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)
combined_tuple = tuple1 + tuple2
print(combined_tuple)
```

```
(1, 2, 3, 4, 5, 6)
```

In [74]:
```
# Bài 13: Tạo tuple có giá trị lặp lại: ('A',) * 5.
repeated_tuple = ('A',) * 5
print(repeated_tuple)
```

```
('A', 'A', 'A', 'A', 'A')
```

In [75]:
```
# Bài 14: Dùng count() để đếm số lần xuất hiện của phần tử trong tuple (1,2,2,3,
my_tuple = (1, 2, 2, 3, 2, 4)
count_of_2 = my_tuple.count(2)
print(count_of_2)
```

```
3
```

In [76]:
```
# Bài 15: Dùng index() để tìm vị trí đầu tiên của phần tử 3 trong tuple (1,3,5,3
my_tuple = (1, 3, 5, 3, 7)
index_of_3 = my_tuple.index(3)
print(index_of_3)
```

```
1
```

In [77]:
```
# Bài 16: Tạo tuple lồng nhau: (('a','b'), (1,2,3)).
nested_tuple = (('a', 'b'), (1, 2, 3))
print(nested_tuple)
```

```
(('a', 'b'), (1, 2, 3))
```

In [78]:
```
# Bài 17: Truy cập phần tử 2 trong tuple con thứ 2 của tuple ở Câu 16.
nested_tuple = (('a', 'b'), (1, 2, 3))
element = nested_tuple[1][1]
print(element)
```

```
2
```

In [79]:
```
# Bài 18: Tạo tuple gồm các số từ 10 đến 15 bằng range() và hàm tuple().
number_tuple = tuple(range(10, 16))
print(number_tuple)
```

```
(10, 11, 12, 13, 14, 15)
```

In [80]:
```
# Bài 19: Dùng unpack để gán tuple (10,20,30) vào 3 biến khác nhau.
my_tuple = (10, 20, 30)
a, b, c = my_tuple
print(a)
print(b)
print(c)
```

```
10
20
30
```

In [81]:
```python
# Bài 20: Thực hành unpack với tuple có nhiều phần tử và dùng * để gom nhóm còn
my_tuple = (1, 2, 3, 4, 5)
a, b, *rest = my_tuple
print(a)
print(b)
print(rest)
```

```
1
2
[3, 4, 5]
```

In [82]:
```python
# Bài 21: In ra tuple đảo ngược từ tuple (1,2,3,4,5).
my_tuple = (1, 2, 3, 4, 5)
reversed_tuple = my_tuple[::-1]
print(reversed_tuple)
```

```
(5, 4, 3, 2, 1)
```

In [83]:
```python
# Bài 22: Tạo tuple rỗng và kiểm tra độ dài.
empty_tuple = ()
print(len(empty_tuple))
```

```
0
```

In [84]:
```python
# Bài 23: Chuyển list [1,2,3] thành tuple.
my_list = [1, 2, 3]
my_tuple = tuple(my_list)
print(my_tuple)
```

```
(1, 2, 3)
```

In [85]:
```python
# Bài 24: Chuyển chuỗi "Hello" thành tuple.
my_string = "Hello"
my_tuple = tuple(my_string)
print(my_tuple)
```

```
('H', 'e', 'l', 'l', 'o')
```

In [86]:
```python
# Bài 25: Tạo tuple gồm các số chẵn từ 0 đến 10.
even_numbers_tuple = tuple(range(0, 11, 2))
print(even_numbers_tuple)
```

```
(0, 2, 4, 6, 8, 10)
```

In [87]:
```python
# Bài 26: Duyệt tuple (2,4,6,8,10) bằng vòng lặp while.
my_tuple = (2, 4, 6, 8, 10)
i = 0
while i < len(my_tuple):
  print(my_tuple[i])
  i += 1
```

```
2
4
6
8
10
```

In [88]:
```python
# Bài 27: So sánh tuple (1,2,3) và (1,2,4).
tuple1 = (1, 2, 3)
```

```
tuple2 = (1, 2, 4)
print(tuple1 < tuple2)
```

```
True
```

In [89]:
```
# Bài 28: In ra phần tử có giá trị lớn nhất trong tuple (1,5,7,3,9).
my_tuple = (1, 5, 7, 3, 9)
max_element = max(my_tuple)
print(max_element)
```

```
9
```

In [90]:
```
# Bài 29: In ra phần tử có giá trị nhỏ nhất trong tuple (2,4,6,1,8).
my_tuple = (2, 4, 6, 1, 8)
min_element = min(my_tuple)
print(min_element)
```

```
1
```

In [91]:
```
# Bài 30: Tính tổng các phần tử trong tuple (1,2,3,4,5) bằng sum().
my_tuple = (1, 2, 3, 4, 5)
total_sum = sum(my_tuple)
print(total_sum)
```

```
15
```

Phần B: Vận dụng (20 câu)

In [92]:
```
# Bài 1: Viết chương trình nhập một chuỗi từ bàn phím và chuyển thành tuple các
input_string = input("Nhập một chuỗi: ")
char_tuple = tuple(input_string)
print(char_tuple)
```

```
Nhập một chuỗi: HONG YEN
('H', 'O', 'N', 'G', ' ', 'Y', 'E', 'N')
```

In [93]:
```
# Bài 2: Viết chương trình nhập một dãy số từ bàn phím (cách nhau bởi dấu cách)
# thành tuple số nguyên.
input_numbers_str = input("Nhập dãy số cách nhau bởi dấu cách: ")
numbers_list_str = input_numbers_str.split()
numbers_list_int = [int(num) for num in numbers_list_str]
numbers_tuple = tuple(numbers_list_int)
print(numbers_tuple)
```

```
Nhập dãy số cách nhau bởi dấu cách: 8 5 78 3 4 5
(8, 5, 78, 3, 4, 5)
```

In [94]:
```
# Bài 3: Viết hàm trả về tuple gồm (giá trị lớn nhất, giá trị nhỏ nhất) của một
def find_min_max(number_tuple):
  if not number_tuple:
    return None
  return (max(number_tuple), min(number_tuple))

my_numbers = (10, 5, 20, 2, 15)
min_max_values = find_min_max(my_numbers)
print(min_max_values)
```

```
(20, 2)
```

In [95]:
```
# Bài 4: Viết chương trình tính số lần xuất hiện của mỗi phần tử trong một tuple
my_tuple = (1, 2, 2, 3, 1, 4, 2, 5)
```

```
frequency = {}
for item in my_tuple:
    if item in frequency:
        frequency[item] += 1
    else:
        frequency[item] = 1
print(frequency)
```

```
{1: 2, 2: 3, 3: 1, 4: 1, 5: 1}
```

In [96]:
```
# Bài 5: Viết chương trình xóa phần tử có giá trị cho trước khỏi một tuple (gợi
# sang list rồi chuyển ngược lại).
def remove_element_from_tuple(my_tuple, element_to_remove):
  my_list = list(my_tuple)
  if element_to_remove in my_list:
    my_list.remove(element_to_remove)
  return tuple(my_list)

my_tuple = (1, 2, 3, 2, 4)
new_tuple = remove_element_from_tuple(my_tuple, 2)
print(new_tuple)
```

```
(1, 3, 2, 4)
```

In [97]:
```
# Bài 6: Viết chương trình gộp hai tuple và sắp xếp tuple kết quả theo thứ tự tă
tuple1 = (5, 2, 8)
tuple2 = (1, 9, 4)
combined_tuple = tuple1 + tuple2
sorted_list = sorted(combined_tuple)
sorted_tuple = tuple(sorted_list)
print(sorted_tuple)
```

```
(1, 2, 4, 5, 8, 9)
```

In [98]:
```
# Bài 7: Viết chương trình tạo tuple từ 2 list: list khóa và list giá trị, rồi g
# của cặp (key,value).
keys = ["name", "age", "city"]
values = ["Alice", 30, "New York"]
if len(keys) == len(values):
    key_value_tuple = tuple(zip(keys, values))
    print(key_value_tuple)
else:
    print("Hai list phải có cùng độ dài.")
```

```
(('name', 'Alice'), ('age', 30), ('city', 'New York'))
```

In [99]:
```
# Bài 8: Viết chương trình tạo tuple gồm các số nguyên tố từ 1 đến 50.
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

prime_numbers = [num for num in range(1, 51) if is_prime(num)]
prime_tuple = tuple(prime_numbers)
print(prime_tuple)
```

```
(2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47)
```

In [100…
```python
# Bài 9: Viết chương trình tìm tất cả các giá trị trùng lặp trong một tuple.
my_tuple = (1, 2, 2, 3, 1, 4, 2, 5)
seen = set()
duplicates = set()
for item in my_tuple:
    if item in seen:
        duplicates.add(item)
    else:
        seen.add(item)
print(tuple(duplicates))
```

```
(1, 2)
```

In [101…
```python
# Bài 10: Viết chương trình tạo tuple gồm các số Fibonacci nhỏ hơn 100.
fib_list = [0, 1]
while fib_list[-1] + fib_list[-2] < 100:
    next_fib = fib_list[-1] + fib_list[-2]
    fib_list.append(next_fib)
fib_tuple = tuple(fib_list)
print(fib_tuple)
```

```
(0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89)
```

In [102…
```python
# Bài 11: Viết hàm nhận vào một tuple số và trả về tuple mới với các phần tử bìn
def square_tuple_elements(number_tuple):
  squared_list = [x**2 for x in number_tuple]
  return tuple(squared_list)

my_numbers = (1, 2, 3, 4, 5)
squared_tuple = square_tuple_elements(my_numbers)
print(squared_tuple)
```

```
(1, 4, 9, 16, 25)
```

In [103…
```python
# Bài 12: Viết hàm hoán đổi vị trí phần tử đầu tiên và phần tử cuối cùng trong m
def swap_first_last(my_tuple):
    if len(my_tuple) < 2:
        return my_tuple
    my_list = list(my_tuple)
    my_list[0], my_list[-1] = my_list[-1], my_list[0]
    return tuple(my_list)

my_tuple = (1, 2, 3, 4, 5)
swapped_tuple = swap_first_last(my_tuple)
print(swapped_tuple)
```

```
(5, 2, 3, 4, 1)
```

In [104…
```python
# Bài 13: Viết hàm đảo ngược một tuple mà không dùng [::-1].
def reverse_tuple_manual(my_tuple):
    reversed_list = []
    for i in range(len(my_tuple) - 1, -1, -1):
        reversed_list.append(my_tuple[i])
    return tuple(reversed_list)

my_tuple = (1, 2, 3, 4, 5)
reversed_tuple = reverse_tuple_manual(my_tuple)
print(reversed_tuple)
```

```
(5, 4, 3, 2, 1)
```

In [105…
```python
# Bài 14: Viết chương trình nhập vào tuple số, in ra tuple chỉ chứa số chẵn.
def get_even_numbers_tuple(number_tuple):
    even_list = [num for num in number_tuple if num % 2 == 0]
    return tuple(even_list)

my_numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
even_tuple = get_even_numbers_tuple(my_numbers)
print(even_tuple)
```

```
(2, 4, 6, 8, 10)
```

In [106…
```python
# Bài 15: Viết chương trình nối nhiều tuple trong một list thành một tuple duy n
tuple_list = [(1, 2), (3, 4), (5, 6)]
single_tuple = ()
for t in tuple_list:
    single_tuple += t
print(single_tuple)
```

```
(1, 2, 3, 4, 5, 6)
```

In [107…
```python
# Bài 16: Viết chương trình đếm số ký tự in hoa và in thường trong tuple ký tự.
char_tuple = ('H', 'e', 'l', 'L', 'o', 'W', 'o', 'r', 'l', 'D')
upper_count = 0
lower_count = 0
for char in char_tuple:
    if char.isupper():
        upper_count += 1
    elif char.islower():
        lower_count += 1
print(f"Số ký tự in hoa: {upper_count}")
print(f"Số ký tự in thường: {lower_count}")
```

```
Số ký tự in hoa: 4
Số ký tự in thường: 6
```

In [108…
```python
# Bài 17: Viết hàm trả về phần tử xuất hiện nhiều nhất trong một tuple.
from collections import Counter

def most_common_element(my_tuple):
    if not my_tuple:
        return None
    counts = Counter(my_tuple)
    most_common = counts.most_common(1)
    return most_common[0][0]

my_tuple = (1, 2, 2, 3, 1, 4, 2, 5)
most_frequent = most_common_element(my_tuple)
print(f"Phần tử xuất hiện nhiều nhất: {most_frequent}")
```

```
Phần tử xuất hiện nhiều nhất: 2
```

In [109…
```python
# Bài 18: Viết hàm tính tích tất cả phần tử trong một tuple.
def multiply_tuple_elements(number_tuple):
    product = 1
    for num in number_tuple:
        product *= num
    return product

my_numbers = (1, 2, 3, 4, 5)
product_result = multiply_tuple_elements(my_numbers)
print(f"Tích các phần tử: {product_result}")
```

```
Tích các phần tử: 120
```

In [110…
```python
# Bài 19: Viết chương trình so sánh 2 tuple và in ra tuple có tổng phần tử lớn h
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)

sum1 = sum(tuple1)
sum2 = sum(tuple2)

if sum1 > sum2:
    print(f"Tuple có tổng lớn hơn là: {tuple1}")
elif sum2 > sum1:
    print(f"Tuple có tổng lớn hơn là: {tuple2}")
else:
    print("Hai tuple có tổng bằng nhau.")
```

```
Tuple có tổng lớn hơn là: (4, 5, 6)
```

In [111…
```python
# Bài 20: Viết chương trình sinh ra tuple gồm các phần tử duy nhất từ tuple ban
my_tuple = (1, 2, 2, 3, 1, 4, 2, 5)
unique_elements = tuple(sorted(set(my_tuple))) # Using sorted to maintain order
print(f"Tuple các phần tử duy nhất: {unique_elements}")
```

```
Tuple các phần tử duy nhất: (1, 2, 3, 4, 5)
```

Bài 51:

- extract: diện tích, số phòng ngủ, số wc, quận, giá

yc: viết hàm, chương trình từ 1 thông báo rao bán bất động sản trên trang batdongsan.com lấy được các thông tin trên 5 tbao khác nhau

In [112…
```python
from bs4 import BeautifulSoup
import re

html_samples = [
    "VÔ CÙNG HỐI TIẾC KHI KHÔNG MUA, THE GIÓ RIVERSIDE QUÁ ĐẸP... 2,29 tỷ · 65,5
    "Bán nhà phố 3 tầng cực đẹp, chỉ 3,5 tỷ · 120 m² · 3 · 3 · Quận 7, TP HCM",
    "Căn hộ cao cấp Sunshine Riverside chỉ từ 2 tỷ · 80 m² · 2 · 2 · Tây Hồ, Hà
    "Nhà mặt tiền trung tâm quận 1, giá 25 tỷ · 95 m² · 4 · 5 · Quận 1, TP HCM",
    "Bán biệt thự ven sông, view cực đẹp 12,5 tỷ · 250 m² · 5 · 6 · Thủ Đức, TP
]

def extract_info(text):
    area = re.search(r"(\d+[\.,]?\d*)\s*m²", text)
    price = re.search(r"(\d+[\.,]?\d*)\s*tỷ", text)
    rooms = re.findall(r"\d+", text)
    bedrooms = rooms[-3] if len(rooms) >= 3 else None
    toilets = rooms[-2] if len(rooms) >= 2 else None
    location = re.findall(r"[A-ZĐÂÊÔƠƯÍÁÉÓÚÝ][^·\d,]+(?:, [A-ZĐÂÊÔƠƯÍÁÉÓÚÝ][^·\d
    return {
        "Giá (tỷ)": price.group(1) if price else None,
        "Diện tích (m²)": area.group(1) if area else None,
        "Phòng ngủ": bedrooms,
        "WC": toilets,
        "Khu vực": location[0].strip() if location else None
    }
```

```python
for sample in html_samples:
    print(extract_info(sample))
```

```
{'Giá (tỷ)': '2,29', 'Diện tích (m²)': '65,56', 'Phòng ngủ': '56', 'WC': '2', 'Kh
u vực': 'Dĩ An, Bình Dương'}
{'Giá (tỷ)': '3,5', 'Diện tích (m²)': '120', 'Phòng ngủ': '3', 'WC': '3', 'Khu vự
c': 'TP HCM'}
{'Giá (tỷ)': '2', 'Diện tích (m²)': '80', 'Phòng ngủ': '80', 'WC': '2', 'Khu vự
c': 'Tây Hồ, Hà Nội'}
{'Giá (tỷ)': '25', 'Diện tích (m²)': '95', 'Phòng ngủ': '4', 'WC': '5', 'Khu vự
c': 'TP HCM'}
{'Giá (tỷ)': '12,5', 'Diện tích (m²)': '250', 'Phòng ngủ': '250', 'WC': '5', 'Khu
vực': 'Thủ Đức, TP HCM'}
```