AI VIET NAM – COURSE 2023

Basic Python – Exercise

Ngày 2 tháng 5 năm 2023

I. Câu hỏi tự luận

Note: Các bạn có thể dùng hàm print để in các variable như sau

```
# Examples
var1 = 'str_var1'
var2 = 3
var3 = 15.5
print('Print single variable (var1): ', var1)
print(f'Print more than one variables: var1={var1}, var2={var2}, var3={var3}')

>> Print single variable (var1):  str_var1
Print more than one variables: var1=str_var1, var2=3, var3=15.5
```

1. **Viết 4 functions thực hiện tính thể tích của các khối sau:**

   - Hình lập phương $V = s^3$ (s là độ dài của cạnh)
     - input function: s
     - output function: V
   - Hình hộp chữ nhật $V = l * w * h$ (l là chiều dài, w là chiều rộng, h: là chiều cao)
     - input function: l, w, h
     - output function: V
   - Hình trụ tròn: $V = \pi * r^2 * h$ (r là bán kính đáy, h là chiều cao)
     - input function: r, h
     - output function: V
   - Hình cầu: $V = \frac{4}{3} * \pi * r^3$ (r bán kính hình cầu)
     - input function: r
     - output function: V

   NOTE: Tùy thuộc vào giá trị π các bạn định nghĩa mà kết quả có thể sai lệch. Ví dụ tự định nghĩa $\pi = 3.14$ hoặc dùng math.pi $\pi = 3.141592653589793$

```
# Examples
calc_cube_volume(s=3)
>> 27

calc_rectangular_prism_volume(l=1, w=2, h=3)
>> 6

```

```
calc_cylinder_volume(r=1.2, h=4)
>> 18.0864

calc_shpere_volume(r=1.3)
>> 9.198106666666666
```

Code Listing 1: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

2. **Viết function thực hiện đánh giá classification model bằng F1-Score.**

   - Precision = $\frac{TP}{TP + FP}$
   - Recall = $\frac{TP}{TP + FN}$
   - F1-score = $2 * \frac{Precision * Recall}{Precision + Recall}$
   - Input: function nhận 3 giá trị **tp, fp, fn**
   - Output: print ra kết quả của **Precision, Recall, và F1-score**

   NOTE: Đề bài yêu cầu các điều kiện sau

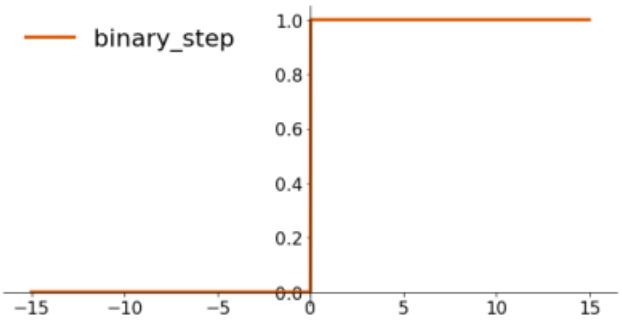
   - Phải **kiểm tra giá trị nhận vào tp, fp, fn là type int**, nếu là type khác thì print ví dụ check fn là float, print **'fn must be int'** và thoát hàm hoặc dừng chương trình.
   - Yêu cầu **tp, fp, fn phải đều lớn hơn 0**, nếu không thì print **'tp and fp and fn must be greater than zero'** và thoát hàm hoặc dừng chương trình

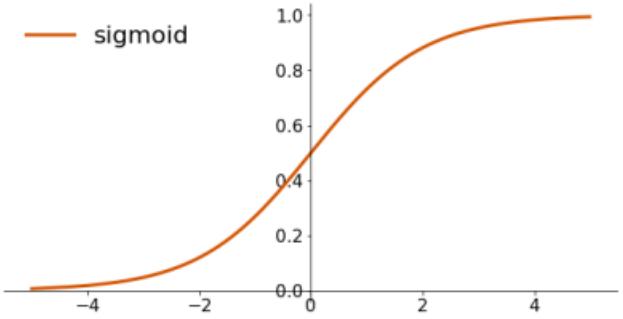
```
# Examples
calc_f1_score(tp=2, fp=3, fn=4)
>> precision is  0.4
recall is  0.3333333333333333
f1-score is  0.3636363636363636

calc_f1_score(tp='a', fp=3, fn=4)
>> tp must be int


calc_f1_score(tp=2, fp='a', fn=4)
>> fp must be int


calc_f1_score(tp=2, fp=3, fn='a')
>> tp must be int

calc_f1_score(tp=2, fp=3, fn=0)
>> tp and fp and fn must be greater than zero

calc_f1_score(tp=2.1, fp=3, fn=0)
>> tp must be int
```

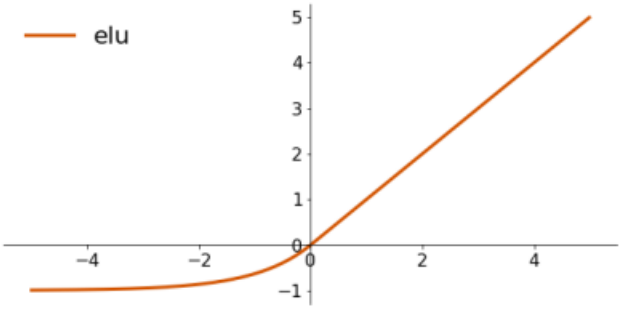
Code Listing 2: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

3. **Viết function mô phỏng theo 3 activation function.**:

- Binary Step Function $f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$
- Sigmoid Function $f(x) = \dfrac{1}{1 + e^{-x}}$
- Elu Function $f(x) = \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$







- Input:
  - Người dùng nhập giá trị **x**
  - Người dùng nhập tên **activation function chỉ có 3 loại (binary, sigmoid, elu)**
- Output: Kết quả **f(x)** (x khi đi qua actiavtion fucntion tương ứng theo activation function name). Ví dụ **nhập x=3, activation_function = 'binary'. Output: print 'binary: f(3)=1'**

NOTE: Lưu ý các điều kiện sau:

- Dùng function **is_number** được cung cấp sẵn để **kiểm tra x có hợp lệ hay không** (vd: x='10', is_number(x) sẽ trả về True ngược lại là False). Nếu **không hợp lệ print 'x must be a number' và dừng chương trình.**
- Kiểm tra **activation function name có hợp lệ hay không nằm trong 3 tên (binary, sigmoid, elu). Nếu không print 'ten_function_user is not supported'** (vd người dùng nhập 'relu' thì print 'relu is not supported')
- Convert **x** sang **float** type

- Thực hiện theo công thức với activation name tương ứng. Print ra kết quả
- Dùng math.e để lấy số e
- $\alpha = 0.01$

```python
# Given
def is_number(n):
    try:
        float(n)   # Type-casting the string to `float`.
                   # If string is not a valid `float`,
                   # it'll raise `ValueError` exception
    except ValueError:
        return False
    return True
```

Code Listing 3: Cho trước hàm is_number

```
exercise3()
>> Input x = 1.5
Input activation Function (binary|sigmoid|elu): sigmoid
sigmoid: f(1.5) = 0.8175744761936437

exercise3()
>> Input x = abc
x must be a number

exercise3()
>> Input x = 1.5
Input activation Function (binary|sigmoid|elu): relu
relu is not supportted
```

Code Listing 4: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

4. **Viết function lựa chọn regression loss function để tính loss:** :
   - MAE = $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
   - MSE = $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
   - RMSE = $\sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$
   - **n** chính là **số lượng samples (num_samples)**, với **i** là mỗi sample cụ thể. Ở đây các bạn có thể hiểu là cứ mỗi **i** thì sẽ **có 1 cặp** y_i **là target và** $\hat{y}$ **là predict**.
   - Input:
     - Người dùng **nhập số lượng sample (num_samples) được tạo ra (chỉ nhận integer numbers)**
     - Người dùng **nhập loss name (MAE, MSE, RMSE-(optional)) chỉ cần MAE và MSE, bạn nào muốn làm thêm RMSE đều được**.
   - Output: Print ra **loss name, sample, predict, target, loss**
     - **loss name:** là loss mà người dùng chọn
     - **sample:** là thứ tự sample được tạo ra (ví dụ num_samples=5, thì sẽ có 5 samples và mỗi sample là sample-0, sample-1, sample-2, sample-3, sample-4)
     - **predict:** là số mà model dự đoán (chỉ cần dùng random tạo random một số trong range [0,10))

- **target:** là số target mà momg muốn mode dự đoán đúng (chỉ cần dùng random tạo random một số trong range [0,10))
- **loss:** là kết quả khi đưa predict và target vào hàm loss
- **note:** ví dụ num_sample=5 thì sẽ có 5 cặp predict và target.

Note: Các bạn lưu ý

- Dùng **.isnumeric() method** để kiểm tra **num_samples** có hợp lệ hay không (vd: x='10', num_samples.isnumeric() sẽ trả về True ngược lại là False). Nếu **không hợp lệ print 'number of samples must be an integer number'** và dừng chương trình.
- **Dùng vòng lặp for, lặp lại num_samples lần. Mỗi lần dùng random modules tạo một con số ngẫu nhiên trong range [0.0, 10.0) cho predict và target**. Sau đó đưa predict và target vào loss function và print ra kết quả mỗi lần lặp.
- **Dùng random.uniform(0,10)** để tạo ra một số ngẫu nhiên trong range [0,10)
- **Giả xử người dùng luôn nhập đúng loss name MSE, MAE, và RMSE (đơn giảng bước này để các bạn không cần check tên hợp lệ)**
- Dùng abs() để tính trị tuyệt đối ví dụ abs(-3) sẽ trả về 3
- Dùng math.sqrt() để tính căn bậc 2

```
exercise4()
>> Input number of samples (integer number) which are generated: 5
Input loss name: RMSE
loss name: RMSE, sample: 0, pred: 6.659262156575629, target: 4.5905830130732355,
    loss: 4.279433398761796
loss name: RMSE, sample: 1, pred: 4.592264312227207, target: 8.447168720237958,
    loss: 14.860287994900718
loss name: RMSE, sample: 2, pred: 8.701801828625959, target: 9.280646891626386,
    loss: 0.3350616069599687
loss name: RMSE, sample: 3, pred: 4.799972972282257, target: 9.877147335937869,
    loss: 25.777699518961764
loss name: RMSE, sample: 4, pred: 0.20159822778697878, target: 5.540221923628147,
     loss: 28.50090296579681
final RMSE: 3.8406610234536727


exercise4()
>> Input number of samples (integer number) which are generated: aa
number of samples must be an integer number
```

Code Listing 5: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

5. Viết các function thực hiện tam giác Pascal và dãy số Fibonacci.

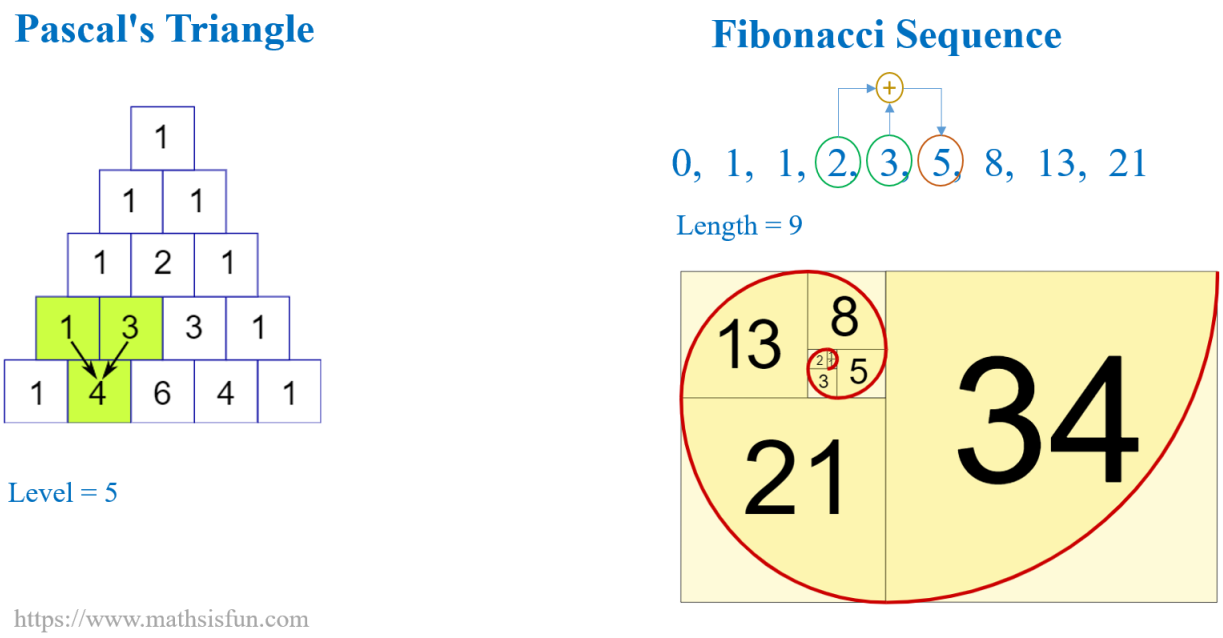
Pascal's Triangle

- Input: level là số lượng hàng
- Output: Print ra kết quả từng hàng theo Pascal's Triangle

Fibonacci Sequence

- Input: length là độ dài của chuỗi Fibonacci
- Output: Print ra kết quả các elemet trong chuỗi Fibonacci

- Các bạn nên tìm hiểu các công thức có thể thực hiện được Pascal's Triangle và Fibonacci Sequence



- Nếu các bạn có kiến thức về list hoặc các kiến thức khác chưa có trong bài học thì các bạn vẫn sử dụng được

NOTE: Các bạn có thể dùng function in như sau để in kết quả trên cùng 1 hàng

```
value1 = 10
value2 = 20
value3 = 30
print(value1, end=' ')
print(value2, end=' ')
print(value3, end=' ')

>> 10 20 30
```

Code Listing 6: Print value trên cùng 1 hàng

```
calc_pas(level=5)
>> 1
1 1
1 2 1
1 3 3 1
1 4 6 4 1

calc_fib(length=9)
>> Fibonacci sequence:
0 1 1 2 3 5 8 13 21
```

Code Listing 7: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

6. Viết 4 functions để ước lượng các hàm số sau.

- Input: x (số muốn tính toán) và n (số lượng bậc muốn xấp xỉ)
- Output: Kết quả ước lượng hàm tương ứng với x. Ví dụ hàm cos(x=0) thì output = 1

NOTE: Các bạn chú ý các điều kiện sau

$$sin(x) \approx \sum_{n=0}^{\infty} (-1)^n \frac{x^{(2n+1)}}{(2n+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots$$

$$cos(x) \approx \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \frac{x^{10}}{10!} + \cdots$$

$$sinh(x) \approx \sum_{n=0}^{\infty} \frac{x^{(2n+1)}}{(2n+1)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \frac{x^9}{9!} + \cdots$$

$$cosh(x) \approx \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + \frac{x^8}{8!} + \frac{x^{10}}{10!} + \cdots$$

- x là radian
- n là số nguyên dương ≥ 0
- các bạn nên viết một hàm tính giai thừa riêng

```
approx_sin(x=3.14, n=10)
>> 0.0015926529267151343

approx_cos(x=3.14, n=10)
>> -0.9999987316527259

approx_sinh(x=3.14, n=10)
>> 11.53029203039954

approx_cosh(x=3.14, n=10)
>> 11.573574828234543
```

Code Listing 8: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

7. Cho một số nguyên dương n, viết phương trình đảo ngược thứ tự các vị trí trong số n. Chỉ dùng while (hay for) và những phép toán cơ bản như +, -, *, /, %, // ...

   - Input: n là một dãy số nguyên dương.
   - Output: Đảo ngược vị trí các số trong n. Ví dụ input: 12345678910, output: 1987654321

   NOTE: Các bạn chú ý các điều kiện sau

   - Không được ép kiểu sang string
   - Chỉ sử dụng while hoặc for loop

```
reverse_number(n=12345678910)
>> 1987654321

reverse_number(n=123456789)
>> 987654321
```

Code Listing 9: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

II. Câu hỏi trắc nghiệm

- Đọc tự luận trước để nắm được idea tổng quát (sẽ không yêu cầu nhưng khuyến khích các bạn tự làm tự luận) và các bài này sẽ được giải trong buổi TA.
- Các bạn phải làm phần trắc nghiệm
  - Các câu hỏi có ký hiệu **(LT)**: [1, 6, 7, 8, 9, 18, 19, 20, 21] là các câu hỏi lý thuyết và đã có sẵn trong file hint nên các bạn chỉ cần hiểu và chạy lại để chọn được đáp án đúng
  - Các câu hỏi có ký hiệu **(Code)**: [2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16, 17, 22, 23, 24, 25, 26] là câu hỏi yêu cầu các bạn phải trực tiếp code vào phần bị khuyết để có thể chọn được đáp án đúng
  - **Lưu ý:** Đối với dạng câu hỏi **(Code)** trong file hint luôn có 1 test casse bắt đầu với từ khóa assert nếu các bạn chạy không báo lỗi có nghĩa các bạn đã vượt qua được test case này và chạy lệnh tiếp theo để trả lời câu hỏi trắc nghiệm
  - **Lưu ý:** Đọc kỹ các code gợi ý và code ví dụ mẫu ở tự luận có thể sẽ có ích cho các bạn khi làm trắc nghiệm

Câu hỏi 1 (LT) : Đầu ra của chương trình sau đây là gì?

```
var1 = 'This is var1'
var2 = 3
var3 = 15.5
print('var1= ', var1)
print(f'var1={var1}, var2={var2}, var3={var3}')
```

a)
var1= This is var1
var1=This is var1, var2=3, var3=15.5
b)
var1= 3
var1= 3, var2= This is var1, var3=15.5
c) None
d) Raise an Error

Câu hỏi 2 (Code): Viết function tính thể tích hình lập phương nhận input là s và return $V = s^3$. Đầu ra của chương trình sau đây là gì?

```
import math
def calc_cube_volume(s):
    # Your code here

    # End your code
assert calc_cube_volume(s=2)==8
print(calc_cube_volume(s=3))
```

a) 1
b) 27
c) 3
d) Raise an Error

Câu hỏi 3 (Code): Viết function tính thể tích hình hộp chữ nhật nhận input là l, w, h và return $V = l * w * h$. Đầu ra của chương trình sau đây là gì?

```
def calc_rectangular_prism_volume(l, w, h):
    # Your code here

    # End your code

assert calc_rectangular_prism_volume(l=2, w=2, h=3)==12
print(calc_rectangular_prism_volume(l=1, w=2, h=3))
```

a) 2
b) 6
c) 36
d) Raise an Error

Câu hỏi 4 (Code): Viết function tính thể tích hình trụ tròn nhận input là r, h và return $V = \pi * r^2 * h$. Đầu ra của chương trình sau đây là gì?

```
import math
PI = 3.14 # math.pi
def calc_cylinder_volume(r, h):
    # Your code here

    # End your code
assert calc_cylinder_volume(r=1.5, h=4) == 28.26
print(round(calc_cylinder_volume(r=1.2, h=4), 2))
```

a) 18.09
b) 9.0
c) 18.0
d) Raise an Error

Câu hỏi 5 (Code): Viết function tính thể tích hình cầu nhận input là r và return $V = \frac{4}{3} * \pi * r^3$. Đầu ra của chương trình sau đây là gì?

```
import math
PI = 3.14 # math.pi
def calc_shpere_volume(r):
    # Your code here

    # End your code

assert round(calc_shpere_volume(r=1), 2) == 4.19
print(round(calc_shpere_volume(r=1.3), 2))
```

a) 9.2
b) 1
c) 10.64
d) Raise an Error

Câu hỏi 6 (LT): Đầu ra của chương trình sau đây là gì?

```
tp = 'true'
if type(tp) != int: # isinstance(tp, int)
        print('tp must be int')
```

a) tp must be int
b) Không có gì xảy ra
c) Raise an Error
d) None

Câu hỏi 7 (LT): Đầu ra của chương trình sau đây là gì?

```
 fp = 1
if type(fp) != int: # isinstance(fp, int)
        print('fp must be int')
```

a) fp must be int
b) Không có gì xảy ra
c) Raise an Error
d) 6

Câu hỏi 8 (LT): Đầu ra của chương trình sau đây là gì?

```
fn = 'One'
if type(fn) != int: # isinstance(fn, int)
    print('fn must be int')
```

a) fn must be int
b) Raise an Error
c) True
d) A and C

Câu hỏi 9 (LT): Đầu ra của chương trình sau đây là gì?

```
tp = 10
fp = -1
fn = 1
if not ((tp > 0) and (fp > 0) and (fn > 0)):
    print('tp and fp and fn must be greater than zero')
```

a) 10
b) -1
c) tp and fp and fn must be greater than zero
d) Raise an Error

Câu hỏi 10 (Code): Viết function thực hiện đánh giá classification model bằng F1-Score. Function nhận vào 3 giá trị **tp, fp, fn** và trả về F1-score

- Precision = $\frac{TP}{TP+FP}$
- Recall = $\frac{TP}{TP+FN}$
- F1-score = $2 * \frac{Precision * Recall}{Precision + Recall}$

Đầu ra của chương trình sau đây là gì?

```
import math
def calc_f1_score(tp, fp, fn):
    # Your code here

```

```
    # End your code

assert round(calc_f1_score(tp=2, fp=3, fn=5), 2) == 0.33
print(round(calc_f1_score(tp=2, fp=4, fn=5), 2))
```

a) 0.33
b) 0.35
c) 0.31
d) Raise an Error

Câu hỏi 11 (Code): Viết function **is_number** nhận input có thể là string hoặc một số **kiểm tra n (một số) có hợp lệ hay không** (vd: n='10', is_number(n) sẽ trả về True ngược lại là False). Đầu ra của chương trình sau đây là gì?

```
import math
def is_number(n):
    # Your code here

    # End your code
assert is_number(3) == 1.0
assert is_number('-2a') == 0.0
print(is_number(1))
print(is_number('n'))
```

a) n
b)
   True
   False
c) 1
d) Raise an Error

Câu hỏi 12 (Code): Viết function thực hiện Binary Step Function $f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$. Nhận input là x và return về kết quả tương ứng trong Binary Step Function (0 hoặc 1). Đầu ra của chương trình sau đây là gì?

```
import math
def calc_bin(x):
    # Your code here

    # End your code
assert calc_bin(4) == 1.0
assert calc_bin(-7) == 0.0
print(calc_bin(2))
print(calc_bin(-1))
```

a) -1
b) 2
c)
   1.0
   0.0
d) Raise an Error

Câu hỏi 13 (Code): Viết function thực hiện Sigmoid Function $f(x) = \dfrac{1}{1+e^{-x}}$. Nhận input là x và return kết quả tương ứng trong Sigmoid Function. Đầu ra của chương trình sau đây là gì?

```python
import math
def calc_sig(x):
    # Your code here

    # End your code

assert round(calc_sig(3), 2)==0.95
print(round(calc_sig(2), 2))
```

a) 0.88
b) 2
c) 0.9907970779778823
d) Raise an Error

Câu hỏi 14 (Code): Viết function thực hiện Elu Function $f(x) = \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$. Nhận input là x và return kết quả tương ứng trong Elu Function. Đầu ra của chương trình sau đây là gì khi $\alpha = 0.01$?

```python
import math
def calc_elu(x):
    # Your code here

    # End your code

assert round(calc_elu(1))==1
print(round(calc_elu(-1), 2))
```

a) -1
b) -0.01
c) -0.106321205588285576
d) Raise an Error

Câu hỏi 15 (Code): Viết function nhận 2 giá trị x, và tên của activation function act_name **activation function chỉ có 3 loại (binary, sigmoid, elu)**, thực hiện tính toán activation function tương ứng với name nhận được trên giá trị của x và trả kết quả. Ví dụ **nhập x=3, activation_function = 'binary'. Output: print 'binary: f(3)=1'**. Đầu ra của chương trình sau đây là gì?

```python
import math
def calc_activation_func(x, act_name):
    # Your code here

    # End your code
assert calc_activation_func(x = 1, act_name='binary') == 1
print(round(calc_activation_func(x = 3, act_name='sigmoid'), 2))
```

a) 0.95
b) 4
c) 1
d) Raise an Error

Câu hỏi 16 (Code): Viết function tính absolute error $= |y - \hat{y}|$. Nhận input là y và $\hat{y}$, return về kết quả absolute error tương ứng. Đầu ra của chương trình sau đây là gì?

```python
def calc_ae(y, y_hat):
    # Your code here

    # End your code
```

```
5
6  y = 1
7  y_hat = 6
8  assert calc_ae(y, y_hat)==5
9  y = 2
10 y_hat = 9
11 print(calc_ae(y, y_hat))
```

a) 7
b) 8
c) 9
d) Raise an Error

Câu hỏi 17 (Code): Viết function tính squared error $= (y - \hat{y})^2$. Nhận input là y và $\hat{y}$, return về kết quả squared error tương ứng. Đầu ra của chương trình sau đây là gì?

```
1  def calc_se(y, y_hat):
2    # Your code here
3
4    # End your code
5  y = 4
6  y_hat = 2
7  assert calc_se(y, y_hat) == 4
8  print(calc_se(2, 1))
```

a) 1
b) 2
c) 3
d) Raise an Error

Câu hỏi 18 (LT): Đầu ra của chương trình sau đây là gì?

```
1 value1 = 10
2 value2 = 20
3 value3 = 30
4 print(value1, end=' ')
5 print(value2, end=' ')
6 print(value3, end=' ')
```

a) 10 20 30
b) 20 30 40
c) 40 50 60
d) Raise an Error

Câu hỏi 19 (LT): Đầu ra của chương trình sau đây là gì?

```
1  def test1(level):
2      for lv in range(level):
3          for i in range(lv + 1):
4              print(i, end=' ')
5          print()
6
7  test1(5)
```

a)
0
0 1
b)
0

1 2
1 2 3

c) Raise an Error

d)
0
0 1
0 1 2
0 1 2 3
0 1 2 3 4

Câu hỏi 20 (LT): Đầu ra của chương trình sau đây là gì?

```
for row in range(4):
  print(row)
```

a)
0
1
2
3

b)
1
2
3

c) 1 2 3 4

d) Raise an Error

Câu hỏi 21 (LT): Đầu ra của chương trình sau đây là gì?

```
def test2(length):
    count = 0
    n1, n2 = 0, 1
    while count < length:
        print(n1, end=' ')
        nth = n1 + n2
        n1 = n2
        n2 = nth
        count += 1
test2(length=8)
```

a) 0 1 1 2 3 5 8 13

b) 1 1 2 3 5 8 13

c) 0 1 1 1 3 5 8 13

d) Raise an Error

Câu hỏi 22 (Code): Dựa vào công thức xấp xỉ cos và điều kiện được giới thiệu các bạn click ở đây. Viết function xấp xỉ cos khi nhận x là giá trị muốn tính cos(x) và n là số bậc muốn xấp xỉ. Return về kết quả cos(x) với bậc xấp xỉ tương ứng. Đầu ra của chương trình sau đây là gì?

```
def approx_cos(x, n):
    # Your code here

    # End your code

assert round(approx_cos(x=1, n=10), 2)==0.54
print(round(approx_cos(x=3.14, n=10), 2))
```

a) 2

b) 1

c) -1.0
d) Raise an Error

Câu hỏi 23 (Code): Dựa vào công thức xấp xỉ sin và điều kiện được giới thiệu các bạn click ở đây. Viết function xấp xỉ sin khi nhận x là giá trị muốn tính sin(x) và n là số bậc muốn xấp xỉ. Return về kết quả sin(x) với bậc xấp xỉ tương ứng. Đầu ra của chương trình sau đây là gì?

```
def approx_sin(x, n):
    # Your code here

    # End your code

assert round(approx_sin(x=1, n=10), 4)==0.8415
print(round(approx_sin(x=3.14, n=10), 4))
```

a) 0.0016
b) 0.0017
c) 0.0018
d) Raise an Error

Câu hỏi 24 (Code): Dựa vào công thức xấp xỉ sinh và điều kiện được giới thiệu các bạn click ở đây. Viết function xấp xỉ sinh khi nhận x là giá trị muốn tính sinh(x) và n là số bậc muốn xấp xỉ. Return về kết quả sinh(x) với bậc xấp xỉ tương ứng. Đầu ra của chương trình sau đây là gì?

```
def approx_sinh(x, n):
    # Your code here

    # End your code

assert round(approx_sinh(x=1, n=10), 2)==1.18
print(round(approx_sinh(x=3.14, n=10), 2))
```

a) 11.53
b) 12.53
c) 13.53
d) Raise an Error

Câu hỏi 25 (Code): Dựa vào công thức xấp xỉ cosh và điều kiện được giới thiệu các bạn click ở đây. Viết function xấp xỉ cosh khi nhận x là giá trị muốn tính cosh(x) và n là số bậc muốn xấp xỉ. Return về kết quả cosh(x) với bậc xấp xỉ tương ứng. Đầu ra của chương trình sau đây là gì?

```
def approx_cosh(x, n):
    # Your code here

    # End your code
assert round(approx_cosh(x=1, n=10), 2)==1.54
print(round(approx_cosh(x=3.14, n=10), 2))
```

a) 11.57
b) 11.58
c) 11.59
d) Raise an Error

Câu hỏi 26 (Code): Cho một số nguyên dương n, viết phương trình đảo ngược thứ tự các vị trí trong số n (input). Chỉ dùng while (hay for) và những phép toán cơ bản như +, -, *, /, %, // ... Output: Đảo ngược vị trí các số trong n. Ví dụ input: 12345678910, output: 1987654321. Không được ép kiểu sang string. Chỉ sử dụng while hoặc for loop. Đầu ra của chương trình sau đây là gì?

```python
def reverse_number(n):
  # Your code here

  # End your code

assert reverse_number(1234)==4321
assert reverse_number(n=34532)==23543
print(reverse_number(12))
print(reverse_number(n=123456789))
```

a)
21
987654321

b)
21112
997654321

c) 123

d) Raise an Error