**ĐỊNH HƯỚNG ÔN CUỐI KÌ 2 MÔN TIN 10**

**I. Trắc nghiệm:**

**“CHÚ Ý: PHẦN LỚN CÂU LỆNH CỦA PYTHON đều viết chữ thường”**

**1. Tìm hiểu lại các khái niệm lập trình, về ngôn ngữ lập trình, , một số ưu điểm của ngôn ngữ lập trình bậc cao so với các loại ngôn ngữ khác**:

1.1 **Khái niệm về lập trình**: Là quá trình sử dụng ngôn ngữ lập trình để hướng dẫn máy tính hiểu và thực hiện giải các bài toán**.**

1.2 **Khái niệm về ngôn ngữ lập trình**: là ngôn ngữ của máy tính dùng để giúp máy tính diễn tả, hiểu và thực hiện được thuật toán.

1.3 **Ưu điểm của ngôn ngữ lập trình bậc cao**: NNLT bậc cao có bảng chữ cái, cú pháp và ngữ nghĩa gần với ngôn ngữ tự nhiên của con người nên dễ hiểu, dễ nhớ và thuận lợi với người học lập trình.

Ví dụ về một số ngôn ngữ lập trình bậc cao: Python, C, C++, PHP….

**2. Tìm hiểu lại: câu lệnh nhập dữ liệu (input), câu lệnh viết dữ liệu (print), câu lệnh gán:**

2.1 **Nhập dữ liệu**: **<tên biến>= input**(“ dòng thông báo nhập dữ liệu”)

----🡪 Viết ra “Dòng thông báo” và dừng lại để nhập giá trị cho **<tên biến>**

**Ví dụ1**: **x=input(“ nhập giá trị của x”)**🡺 câu lệnh nó có ý nghĩa: viết ra dòng thông báo: ***nhập giá trị của x***  sau đó đợi để nhập giá trị cho x (x này nhập vào có kiểu dữ liệu là kiểu xâu)

**Ví dụ 2: x= int(input(“ nhập giá trị của x”))**🡺 câu lệnh nó có ý nghĩa: viết ra dòng thông báo: ***nhập giá trị của x*** sau đó đợi để nhập giá trị cho x (x này nhập vào có kiểu dữ liệu là kiểu số nguyên)

2.2 **Viết dữ liệu:**  print(Dữ liệu)

-----🡪 viết Dữ liệu ra màn hình ( Dữ liệu có thể là: giá trị (biến, hằng, biểu thức) hoặc là dòng thông báo)

**Ví dụ1**: print(3+5)=🡺 viết giá trị 8 ra màn hình (do 3+5=8)

**Ví dụ2**: print(“ nói mãi mà không nhớ”)🡺 viết thông báo: Nói mãi mà không nhớ

2.3 **Câu lệnh gán giá trị:**  **Biến=<biểu thức>**

-----🡪 tính giá trị của <biểu thức> sau đó gán giá trị đó cho **Biến**

**Ví dụ1**: a=5+6 🡪 lệnh này gán giá trị 11 (5+6=11) cho biến a

1. **Tìm hiểu lệnh rẽ nhánh: if….else (dạng thiếu, dạng đủ):**

**3.1 Dạng thiếu**: if <điều kiện>: **<khối lệnh)**

--……🡪 :

+ khi <điều kiện> có giá trị True (đúng) thì thực hiện <**khối lệnh>**

+ khi <điều kiện> có giá trị False (sai) thì bỏ qua <**khối lệnh>**

**ví dụ1**:

a=4

b= 3

T=0

if a>b: T=a+b

print(T)

-🡪 T sẽ có giá trị là 7 vì: < điều kiện> là a>b có giá trị **đúng** ( 4>3 có giá trị **True**) nên sẽ thực hiện **<khối lệnh>** là câu lệnh T=a+b (tức T= 3+4 =>T=7), lệnh prin(T): là viết giá trị T =7 ra màn hình

**ví dụ2**:

a=3

b= 4

T=0

if a>b: T=a+b

print(T)

-🡪 T sẽ có giá trị là 0 vì: < điều kiện> là a>b có giá trị **False**( 3>4 có giá trị **False**) nên sẽ **bỏ qua** **<khối lệnh>** là câu lệnh T=a+b, lệnh prin(T): là viết giá trị T =0 (vì không có câu lệnh nào làm biến đổi giá trị của T nên T vẫn giữ giá trị là T=0) ra màn hình

**3.2 Dạng đủ**: if <**điều kiện**>: <**khối lệnh 1>**

else: <**khối lệnh2>**

-------🡪:

+ khi <**điều kiện**> có giá trị True (đúng) thì thực hiện <**khối lệnh1>** bỏ qua<**khối lệnh2> .**

+ khi <**điều kiện**> có giá trị False (sai) thì thực hiện <**khối lệnh2>** bỏ qua<**khối lệnh1> .**

**Ví dụ**

a=3

b= 4

T=0

if a>b: T=a+b

else: T=a-b

print(T)

=🡺 T có giá trị là = -1 vì <điều kiện> có giá trị **False** ( vì 3>4 có giá trị False) nên thực hiện **<Khối lệnh 2>** là T=a-b (3-4 có giá trị là -1) và bỏ qua **<Khối lệnh 1>** là T=a+b.

1. **TÌm hiểu cấu trúc lặp: Lặp với số lần biết trước (for….), lặp với số lần chưa biết trước (while…):**

4.1 **Cấu trúc lặp với số lần biết trước** (for….):

**for biến\_chạy in range (m,n):**

**Khối lệnh cần lặp**

--🡪 **biến\_chạy** sẽ nhận lần lượt các giá trị trong phạm vi lần lượt từ **m** đến

**n-1** ( **hàm range(m,n):** là hàm sinh ra **một miền giá trị:** từ **m tới n-1**), tương ứng với ***mỗi giá trị*** của **biến\_chạy** sẽ thực hiện **Khối lệnh cần lặp** 1 lần (tức trong toàn bộ quá trình: **Khối lệnh cần lặp** sẽ thực hiện n-m lần tất cả)

**ví dụ: for i in range(1,11):**

**print(“ Love DH”)**

-🡪 viết “Love DH” 10 lần

Giải thích:

**biến\_chạy** ở đây là **i** sẽ nhận lần lượt các giá trị từ: 1,2,3….10 (***vì do range(1,11) : sinh ra miền giá trị từ: 1,2,3,4…..9,10***)

**khi i =1 thì thực hiện print(“ Love DH”)**

**khi i =2 thì thực hiện print(“ Love DH”)**

**khi i =3 thì thực hiện print(“ Love DH”)**

**…..**

**khi i =10 thì thực hiện print(“ Love DH”)**

(Tổng là 10 lần **print(“ Love DH”): tức là viết 10 lần Love DH ra màn hình)**

4.2 **Cấu trúc lặp với số lần chưa biết trước** (while….):

**while** *<điều kiện>:*

*<khối lệnh>*

-🡪 Trong khi **<điều kiện>:** vẫn còn giá trị **True** thì tiếp tục thực hiện **<khối lệnh>** (tức là **<khối lệnh>** được lặp lại nhiều lần khi **<điều kiện>** vẫn có giá trị True (đúng)), khi **<điều kiện>** có giá trị **False** thì thoát ra (không thực hiện **<khối lệnh>** nữa).

Ví dụ:

***i= 1***

***while i<=4:***

***print(“chào DH”***

***i=i+1***

* Chương trình này viết 4 lần “ chào DH”

Giải thích:

+ với i =1 thì <điều kiện> là i<=4 có giá trị **True** ( 1<=4 cho giá trị True) nên sẽ thực hiện <khối lệnh> (gồm 2 lệnh):

print(“ chào DH”) -🡪 viết “ chào DH 1 lần **(lần 1)**

i=i+1 ( lúc này i=1+1, i có giá trị =2)

sau đó quay trở về kiểm tra <điều kiên>:

+ với i =2 thì <điều kiện> là i<=4 có giá trị **True** ( 2<=4 cho giá trị True) nên sẽ thực hiện <khối lệnh> (gồm 2 lệnh):

print(“ chào DH”) -🡪 viết “ chào DH 1 lần **(lần 2)**

i=i+1 ( ( lúc này i=2+1, i có giá trị =3)

sau đó quay trở về kiểm tra <điều kiên>:

+ với i =3 thì <điều kiện> là i<=4 có giá trị **True** ( 3<=4 cho giá trị True) nên sẽ thực hiện <khối lệnh> (gồm 2 lệnh):

print(“ chào DH”) -🡪 viết “ chào DH 1 lần **(lần 3)**

i=i+1 ( ( lúc này i=3+1, i có giá trị =4)

sau đó quay trở về kiểm tra <điều kiên>:

+ với i =4 thì <điều kiện> là i<=4 có giá trị **True** ( 4<=4 cho giá trị True) nên sẽ thực hiện <khối lệnh> (gồm 2 lệnh):

print(“ chào DH”) -🡪 viết “ chào DH 1 lần **(lần 4)**

i=i+1 ( ( lúc này i=4+1, i có giá trị =5)

sau đó quay trở về kiểm tra <điều kiên>:

+ với i =5 thì <điều kiện> là i<=4 có giá trị **False** ( 5<4 cho giá trị False) nên sẽ không thực hiện <khối lệnh> nữa và thoát ra

==🡺 Tổng là thực hiện <khối lệnh> 4 lần (viết ra 4 lần “chào DH”)

1. **Tìm hiểu chương trình con: cú pháp viết ctc, cách gọi ctc:**

5.1 Cú pháp viết chương trình con: **def** *tên\_hàm (tham số)***:**

***Các lệnh mô tả hàm***

***Tên\_hàm: do chúng ta tự đặt theo quy tắc đặt tên (đúng ý nghĩa của bài toán đang giải)***

***Tham\_số: là đầu vào giả định của bài toán (tham số hình thức): nó dạng như là các thông số đầu vào giả định (coi như chỉ chiếm vị trí: chưa có giá trị cụ thể)***

***Các lệnh mô tả hàm: là các lệnh để giải bài toán (chương trình con =hàm)***

***2 loại chương trình con:***

***+ctc trả về giá trị: cuối ctc thường có câu lệnh return <giá trị trả về>***

***+ ctc không trả về giá trị: thì cuối ctc ko có câu lệnh return <giá trị trả về>***

***Ví dụ1: xây dựng chương trình con tính diện tích tam giác vuông:***

***def S\_tgv(a,b):***

***s=1/2\*(a\*b)***

***return s***

***S\_tgv(3,4)***

***S\_tgv(6,8)***

**def** *tên\_hàm (tham số)***:**

***Các lệnh mô tả hàm***

***Ví dụ 2: Xây dựng chương trình con viết 10 lần “ xin chào DH”***

***def viet\_10\_lan():***

***for i in range(1,11):***

***print(“xin chào DH”)***

***viet\_10\_lan***

***viet\_10\_lan***

***viet\_10\_lan***

* ***Lời chương trình con: Tên\_ham (<tham số thực sự>):***

5.2 Cách gọi chương trình con:

+nếu ctc có tham số thực sự : **<tên\_hàm> (<tham số thực sự>)** (

+ nếu ctc không có tham số thực sự thì : <tên\_hàm>

1. **TÌm hiểu dữ liệu kiểu xâu: 5 câu hỏi**
   1. **Khái niệm xâu: là dãy các kí tự nằm trong bảng mã ASCII**
   2. **Các hàm xử lý xâu:**

**+ Hàm y.count(x): đếm số lần xuất hiện (không giao nhau của xâu x trong xâu y)**

**Ví dụ:**

**y= “abca”**

**x= “a”**

**c=y.count(x)**

**d=y.count(“b”)**

**Đối với chương trình trên:**

**+ c có giá trị là 2: vì tìm thấy 2 xâu x ( x là kí tự “a”) trong xâu y**

**+ d có giá trị là 1: vì tìm thấy 1 kí tự “b” trong xâu y**

**+ Hàm len(y): cho số lượng kí tự trong xâu y**

**Ví dụ:**

**y= “Dam Ha”**

**c=len(y)**

**Đối với chương trình trên:**

**+ c có giá trị là 6: vì trong xâu y có 6 kí tự (kể cả dấu cách)**

**+ Hàm y.find(x): tìm vị trí xuất hiện đầu tiên của xâu x trong xâu y, nếu không tìm thấy thì trả kết quả là -1**

**Ví dụ:**

**y= “Dam Ha”**

**c=y.find(“a”)**

**Đối với chương trình trên:**

**+ c có giá trị là 1: vì trong xâu y kí tự “a” xuất hiện lần đầu tại vị trí số 1**

**+ Hàm y.replace(x1,x2): thay thế xâu x1 trong y bằng xâu x2**

**Ví dụ:**

**y= “Dam Ha”**

**c=y.replace(“a”,“e”)**

**Đối với chương trình trên:**

**+ c là xâu: “Dem He” vì: các kí tự “a” trong xâu y sẽ được thay thế bằng kí tự “e”**

1. **TÌm hiểu dữ liệu kiểu danh sách:** 
   1. **Các cách tạo danh sách: có 3 cách tạo**

**+ Cách 1: dùng phép gán: gán trực tiếp các giá trị cho danh sách**

**Ví dụ: A=[1,3,7]**

**+ Cách 2: dùng câu lệnh for để gán giá trị trong khoảng cho trước:**

**Ví dụ : B= [i for i in range(3,6)]----🡪 danh sách B = [3,4,5]**

**+ Cách 3: Khởi tạo danh sách số nguyên hay số thực từ dữ liệu nhập vào:**

**+ số nguyên:** a= [int(a) for i in input().split()]

**+ số thực:** a= [float(a) for i in input().split()]

* 1. **Cách truy cập đến các phần tử trong danh sách: tên\_ds[vị trí]**

Ví dụ: a=[1,5,7,9]

c= a[3]

print(a[1])

Trong chương trình trên:

+ biến c có giá trị bằng 9 ( tại vì c=a[3] có nghĩa là c có giá trị bằng giá trị ở vị trí số 3 của danh sách a

+ lệnh print(a[1])-🡪 viết giá trị 5 ra màn hình vì a[1] có giá trị bằng 5

***(chú ý: trong kiểu danh sách vị trí khởi đầu là vị trí 0)***

* 1. **Các hàm xử lý danh sách:**

**+ Hàm: a.append(x): bổ sung phần tử x vào cuối danh sách a**

**Ví dụ:**

a=[1,5,7]

a.append(6)

Trong chương trình trên:

Danh sách a đã được thay đổi thành a= [1,5,7,6] vì phần tử **6** được bổ sung vào cuối danh sách a bởi câu lệnh a.append(6)

**+ Hàm: a.pop(i): lấy ra phần tử ở vị trí i của danh sách a sau đó xóa phần tử này trong danh sách**

**Ví dụ:**

a=[1,5,7,6]

c=a.pop(3)

print(a)

Trong chương trình trên:

+ c có giá trị là 6: c được tạo ra bằng cách lấy phần tử ở vị trí số 3 của danh sách a ( có giá trị bằng 6)

+ Danh sách a đã được thay đổi thành a= [1,5,7] vì phần tử ở vị trí số 3 có giá trị **6** đã được lấy ra

**+ Hàm: a.insert(i,x): chèn phần tử có giá trị là x vào vị trí i của danh sách a**

**Ví dụ:**

a=[1,5,7]

a.insert(1,6)

Trong chương trình trên:

Danh sách a đã được y đổi thành a= [1,6,5,7] vì phần tử có giá trị **6** được chèn vào vị trí số 1 của danh sách a bởi lệnh: a.insert(1,6)

**+ Hàm: a.sort(): sắp xếp danh sách a theo chiều không giảm**

**Ví dụ:**

a=[5,1,7]

a.sort()

Trong chương trình trên:

Danh sách a đã được thanh đổi thành a= [1,5,7] vì danh sách a đã được sắp xếp lại bởi câu lệnh: a.sort().

* 1. **Phép ghép danh sách: sử dụng phép +**

**Ví dụ**

**a=[1,3]**

**c=[2,6]**

**d=a+c**

**e=c+a**

Trong đoạn lệnh trên:

+ danh sách **d** có cấu tạo là: **[1,3,2,6]**

+ danh sách **e** có cấu tạo là: **[2,6,1,3]**

**(Chú ý vị trí của danh sách đối với phép +)**

1. **Kiểm thử và gỡ lỗi chương trình: 3 loại lỗi:**

+ Lỗi cú pháp: là lỗi câu lệnh viết không theo quy định của ngôn ngữ lập trình.

+ Lỗi ngoại lệ: còn gọi là lỗi **Runtime**, lỗi này xảy ra khi chương trình đang chạy, một lệnh nào đó không thể thực hiện được.

+ Lỗi ngữ nghĩa: là lỗi mặc dù các câu lệnh viết đúng quy định của ngôn ngữ lập trình nhưng sai một số thao tác nào đó (dẫn đến cho kết quả sai)

1. **Các bước giải bài toán trên máy tính: 4 bước:**

+ B1: Xác định bài toán

+ B2: Tìm thuật toán giải bài toán và cách tổ chức dữ liệu

+ B3: Viết chương trình

+ B4: Kiểm thử và hiệu chỉnh chương trình

1. **Tìm hiểu về nghề thiết kế lập trình: nghề thiết kế và lập trình gồm 4 quá trình:**

+ Phân tích hệ thống

+ Thiết kế phần mềm

+ Lập trình

+ Kiểm thử phần mềm

1. **Tự luận:**

**Câu 1** (1đ). Tìm hiểu lại về ngôn ngữ lập trình: khái niệm lập trình, ngôn ngữ lập trình, ưu điểm của ngôn ngữ lập trình bậc cao , tìm hiểu tên một số loại ngôn ngữ lập trình bậc cao.

***(trình bày ở mục 1 phần trắc nghiệm—chỉ việc đọc lại)***

**Câu 2** (1đ) Xây dựng 1 chương trình con đơn giản: Ví dụ Tính diện tích hình tròn, diện tích hình vuông, hình chữ nhật, hình thang, tam giác vuông, tam giác thường.

+ hình tròn: def Hinh\_tron (R):

S=3.14\*R\*R

return S

+ hình vuông: def Hinh\_vuong(a):

S=a\*a

return S

+ hình chữ nhật, hình thang, **tam giá vuông**, tam giác thường: **tự viết**

***Câu 3***. Viết một chương trình hoàn chỉnh để giải một bài toán đơn giản:

Ví dụ:

**Bài 1**. Cho số số nguyên dương N (N>=2) hãy lập trình để nhập số nguyên dương N từ bàn phím sau đó tính tổng bình phương các số chẵn trong phạm vi từ 1 đến N sau đó viết kết quả ra màn hình:

***N=int(input(“ Nhập số nguyên dương N”))***

***T=0***

***for i in range (1,N+1):***

***if i%2==0: T=T+i\*i***

***print(T)***

**Bài 2**. Viết chương trình thông báo giải thưởng dựa trên kết quả học tập của một học sinh (ĐTB), với ĐTB được nhập từ bàn phím, quy định tính giải như sau:

+ Học sinh giỏi: có ĐTB>=8.0 giải thưởng là 1 triệu

+ Học sinh khá: có 6.5=<ĐTB<8.0 giải thưởng là 500k

+ Học sinh có ĐTB<6.5 không có giải thưởng

Giải:

**DTB=float(input(“ Nhập điểm trung bình của bạn: ”))**

**if DTB>=8.0: print(“ Bạn đạt giải thưởng 1 triệu”)**

**if (DTB>=6.5)and (DTB<8.0): print(“ Bạn đạt giải thưởng 500k”)**

**if DTB<6.5: print(“ Bạn chưa đạt giải”).**

**Bài 3**. Viết chương trình tính số tiền phải thanh toán dựa trên số lượng hàng nhập vào (được nhập từ bàn phím với số lượng nhập tùy ý):

**+ số hàng nhập: sohang <=100: tính giá 50đ/1 hàng**

**+ số hàng nhập: 100<sohang <=200: 100 hàng đầu tính giá 50đ, số còn lạ tính 100đ (ví dụ số hàng: 139 tiền được tính như sau: sotien= 100\*50+39\*100)**

+ Số hàng nhập: sohang>200: 100 số đầu tính 50đ, 100 số sau tính 100đ, số còn lại tính 150đ ( ví dụ số hàng nhập là 219 thí sotien= 100\*50+100\*100+19\*150).

***Chương trình này tự viết nhá. kkkk***