THỰC HÀNH TOÁN CAO CẤP

*TÀI LIỆU PHỤC VỤ SINH VIÊN NGÀNH KHOA HỌC DỮ LIỆU*

Nhóm biên soạn: TS. Hoàng Lê Minh – Khưu Minh Cảnh – Hoàng Thị Kiều Anh – Lê Thị Ngọc Huyên – …

TP.HCM – Năm 2019

**MỤC LỤC**

[CHƯƠNG 7: DÃY, CHUỖI SỐ VÀ ỨNG DỤNG 3](#_Toc18953)

[1. Một số lưu ý khi sử dụng sympy (phần 1) 3](#_Toc18954)

[2. Hàm số, sai phân và tích phân số 4](#_Toc18955)

[1.1. Các tích phân không tính được 4](#_Toc18956)

[1.2. Chỉ số CO về cung lượng tim – Cardinal Output 6](#_Toc18957)

[3. Giới thiệu một số ứng dụng của đường cong 8](#_Toc18958)

[2.1. Tính toán độ dài đường cong - ArcLength 8](#_Toc18959)

[2.2. Bài toán “đường cong mềm mại” 11](#_Toc18960)

[4. Xử lý dãy số và chuỗi số trong thư viện sympy 13](#_Toc18961)

[a. Dãy số trong sympy 13](#_Toc18962)

[b. Chuỗi số trong sympy 15](#_Toc18963)

[BÀI TẬP CHƯƠNG 7 17](#_Toc18964)

# CHƯƠNG 7: DÃY, CHUỖI SỐ VÀ ỨNG DỤNG

**Mục tiêu:**

* *Tích phân, dãy số (sequence), sai phân và tích phân số.*
* *Ứng dụng một số “kết quả” đối với đường cong;*
* *Chuỗi số (series) và ứng dụng*

**Nội dung chính:**

# 1. Một số lưu ý khi sử dụng sympy (phần 1)

**Dưới đây là một số lưu ý khi sử dụng sympy mà người sử dụng cần biết:**

* SymPy chạy trên nền ngôn ngữ lập trình Python, do đó, nhiều thứ có khác so với trong các hệ đại số trên máy tính độc lập như Maple hoặc Mathematica.
* **C, O, S, I, N, E,** và **Q** là những biến đặc biệt riêng; tốt nhất, chúng ta không nên sử dụng những biến đó. Ví dụ, **I** đại diện cho đơn vị ảo và **E** đại diện cho số Euler.
* SymPy, giống như Python, mọi lệnh nhân phải tường minh, nghĩa là không thể ghi 2x mà phải

ghi là **2\*x**.

* Chúng ta có thể chuyển đổi bất kì chuỗi nào vàp trong biểu thức sử dụng hàm **sympify**(). Điều này sẽ tự động định nghĩa biến cho chúng ta. Ví dụ: chúng ta chỉ việc nhập lệnh **sympify("a^2**

**+ cos(b)")**và nó sẽ hoạt động.

* Chúng ta có thể lưu trữ các giá trị đặc biệt trong các biến thường để chúng ta có thể không cần phải thêm ".**evalf()**" mỗi khi sử dụng biến. Ví dụ: **x=pi.evalf()** hoặc **y=E.evalf()**.
* Phương pháp tốt nhất để kiểm tra sự bằng nhau là sử dụng hàm simplify để kiểm tra sự khác nhau giữa hai biểu thức như thế nào. Ví dụ, để kiểm tra sự bằng nhau của (x-1)\*\*2 và x\*\*2 -

2\*x + 1, chúng ta chỉ cần lệnh **>>> print simplify((x-1) \*\* 2 - (x\*\*2 - 2\*x + 1))** và xem kết quả nếu bằng 0. Nếu chúng ta nghi ngờ biểu thức bằng 0 thì chúng ta có thể khẳng định lần nữa bằng lệnh expr.equals(0); câu trả lời False nếu biểu thức không bằng 0 (không thể) và **None** nếu nó không thể quyết định (lúc bằng lúc không).

* Một số hàm SymPy được đặt tên khác với tên trong hệ thống. Cụ thể, các hàm đảo trong SymPy là asin, acos, và atan mà không phải là arcsin/arccos/arctan.

# 2. Hàm số, sai phân và tích phân số

Dãy số là danh sách các số theo thứ tự. Sai phân là hiệu của các số liền nhau trong dãy số. Các ứng dụng trong thực tế thường là những dãy số rời rạc. Từ đó, chúng ta tìm cách xây dựng hàm số của chúng để tính tích phân. Một cách khác, chúng ta có thể sử dụng một số phương pháp tính tích phân số như Simpson để tính toán tích phân cho một dãy số.

Tuy vậy, khi lấy tích phân, với một số hàm số, chúng ta cần phải xử lý (như tính trên từng khoảng, biến đối x thành y, tính các giới hạn…) hoặc chứng minh không tồn tại tích phân (giá trị là vô cùng hoặc không xác định).

### 1.1. Các tích phân không tính được

Một số tích phân sẽ phân kỳ, nghĩa là khi tính toán giá trị sẽ là vô cùng. Xét các ví dụ sau: **Thực hành 1: Tích phân không xác định loại 1**

1

= log 2 −

>>> from sympy import Symbol, Integral

>>> x = Symbol('x')

>>> f = 1.0/x

>>> unbound1 = Integral(f, (x, 0, 2))

>>> unbound1.doit()

………………………oo…………………………..  sinh viên điền giá trị

**Thực hành 2: Tích phân không xác định loại 2:**

Nghĩa là chúng ta không thể kiểm soát được giá trị tích phân

1

>>> from sympy import Symbol, Integral

>>> x = Symbol('x')

>>> f = 1.0/x\*\*2

>>> unbound2 = Integral(f, (x, -1, 1))

>>> unbound2.doit()

……………………… oo……………………..  sinh viên điền giá trị

**Thực hành 3: Tích phân không liên tục**

| |

>>> from sympy import Symbol, Integral, Abs

>>> x = Symbol('x')

>>> f = Abs(x)

>>> discontinous1 = Integral(f, (x, -1, 2))

>>> discontinous1.doit()

……………………… 5/2………………………..  sinh viên điền giá trị

**Thực hành 4: Tích phân có hàm thỏa không tồn tại trong miền xác định (thực)**

>>> from sympy import Symbol, Integral

>>> x = Symbol('x')

>>> f = x\*\*(0.5)

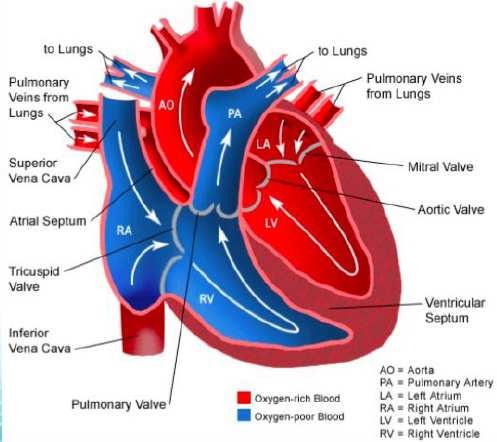
>>> undef1 = Integral(f, (x, -1, 1))

>>> undef1.doit()

……… 0.666666666666667 + 0.666666666666667\*I……….  sinh viên điền giá trị

### 1.2. Chỉ số CO về cung lượng tim – Cardinal Output

Huyết áp – Blood Pressure được chia thành 2 chỉ số áp lực trong máu: **tâm thu** (systolic) và **tâm trương** (diastolic). Bình thường chỉ số đọc được là 120/80 (đơn vị là mg thủy ngân). Vấn đề: làm cách nào có những chỉ số đo đó để chúng ta có thể can thiệp vào chúng như thế nào? Hoặc câu hỏi đơn giản sự chính xác của thiết bị đo như thế nào giả định ta biết được phương pháp đo của thiết bị?



Cung lượng tim nói nôm na là chỉ số áp lực máu phản xạ theo lực các vành động mạch (arterial walls). Áp lực này tạo thành do nhịp đập từ tim trong Chu trình tuần hoàn (Cardiac cycle) của máu. Với toán học, chúng ta có thể sử dụng tích phân để tìm ra cung lượng tim (lượng máu tim bơm) trong một khoảng thời gian nhất định. Công thức được gọi là CO (cardiac output) với kỹ thuật sử dụng phương thức tiêm thuốc nhuộm (dye-dilution) như sau:

=

Trong đó:

1. là lượng dòng (gọi là flow hoặc flux), là lượng máu bơm của tim đơn vị tính là lít máu/phút.
2. là lượng thuốc nhuộm tiêm vào.
3. [0, ] là thời gian đo iv. là hàm mà mô hình lượng thuốc nhuộm còn lại sau thời gian t.

Về phương pháp:

+ Phương pháp tiêm thuốc nhuộm được thực hiện bằng cách tiêm vào bệnh nhân một loại thuốc để nó chạy vào ngay trong tâm nhỉ phải (right atrium) của tim. Theo đó, dòng nhuộm cùng với máu vào trong động mạch chủ (aorta). Một thiết bị được đưa vào trong động mạch chủ để đo mức độ tập trung của thuốc nhuộm khi ra khỏi tim trong những khoảng thời gian bằng nhau [0, ] (đến khi hết hoàn toàn thuốc). Gọi là mức độ tập trung của thuốc nhuộm tại thời điểm . Bằng cách chia nhỏ khoảng [0, ] thành các Δ , lượng thuốc nhuộm chảy qua điểm đo trong suốt thời gian từ =# đến =# là:

$ứ& độ ậ\* +,-. ạ0 ℎờ0 đ0ể$ 4ℎố0 6ượ-. = & # Δ

Với là tỉ lệ dòng được tính. Tổng số thuốc nhuộm là:

> >

9 c t< FΔ = F 9 c t<Δ

<? <?

Khi , lượng thuốc nhuộm là:

D

A = F c t dt

Và từ công thức trên, ta tính được F vì lượng thuốc nhuộm A là biết trước. Trong thực tế, sai số của phương pháp này bắt buộc dưới 5%.

Tài liệu tham khảo thêm: https://en.wikipedia.org/wiki/Cardiac\_output

**Thực hành 5: Một máy đo đo được lượng thuốc nhuộm trong máu với số liệu như sau:**

Thời gian 0 8

2

4

6

10

12

14

16

C(t)

0.0

4.1

8.9

8.5

6.7

4.

3

2.5

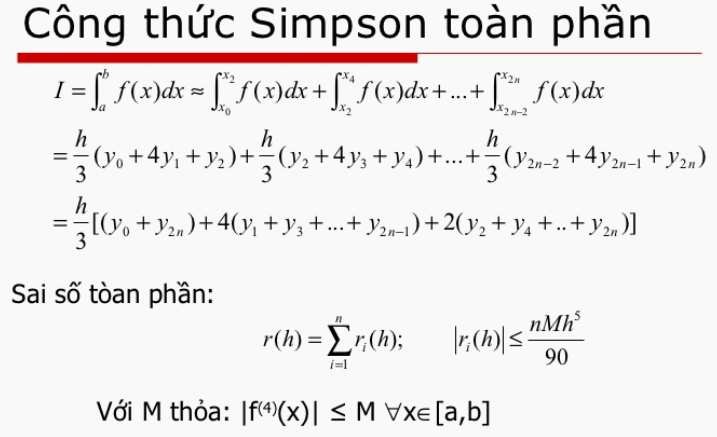
1.2

0.2

Hãy sử dụng ngôn ngữ Python để:

Tính giá trị F (lít máu/phút), giả định A=5.5.

Gợi ý: Sử dụng một phương pháp như Simpson để tính tích phân số C(t)



Giải:

Gói **scipy** hỗ trợ việc tính tích phân số. Gói **integrate** có lệnh tính tích phân số bằng phương pháp simpson thông qua lệnh **simps** như sau:

>>> from **scipy** import **integrate**

>>> import **numpy** as **np**

>>> y = np.array([0.0, 4.1, 8.9, 8.5, 6.7, 4.3, 2.5, 1.2,0.2])

>>> x = np.arange(0,9)

>>> **integrate.simps(y, x)**

…………… 36.26666666666666…………………  sinh viên điền vào

Vì cứ mỗi 2 giây nên trong công thức tính Simpson, ∆ = 2. Do đó chúng ta phải nhân 2 kết quả:

>>> S = 2\*integrate.simps(y, x)

……………… 72.53333333333332………………  sinh viên điền vào

Từ đó, ta có cung lượng tim là: = F = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ lít máu/giây, nghĩa là tim sẽ bơm:

G

\_\_\_\_\_\_\_\_\_\_ lít máu/phút. Thực hiện bằng Python các câu lệnh:

>>> A = 5.5

>>> F = A/S \* 60

……………… 4.5496323529411775……………  sinh viên điền vào

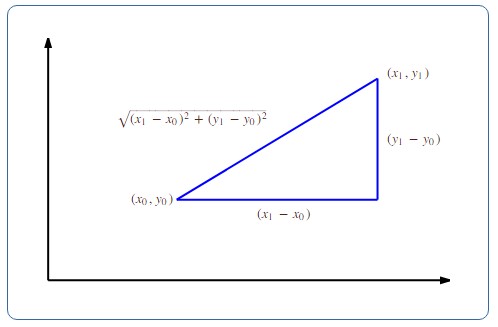
# 3. Giới thiệu một số ứng dụng của đường cong

Phần này giới thiệu một số kết quả cơ bản của đường cong. Các kết quả này là từ các lý thuyết về tích phân, đạo hàm của hàm một và nhiều biến.

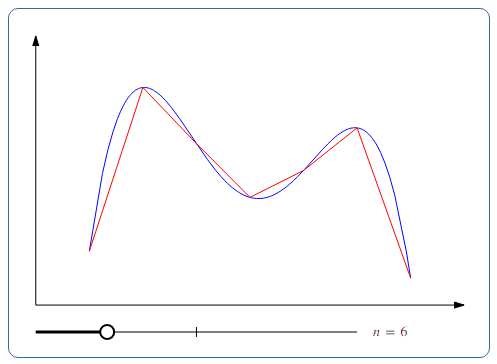
### 2.1. Tính toán độ dài đường cong - ArcLength

Dưới đây là ứng dụng hình học của tích phân: tìm độ dài của một phần đường cong.

Chúng ta đã biết việc tính toán độ dài của một đoạn thẳng. Nếu 2 điểm đầu cuối (gọi là endpoints) là I , J và I , J khi đó độ dài giữa hai điểm sẽ là K − + J − J theo định lý Pythagore (bên dưới là hình minh họa độ dài đoạn thẳng)



Tuy nhiên, nếu đồ thị hàm M không phải là đoạn thẳng thì chúng ta cần công cụ toán học để đo. Giả định hàm M là khả vi thì chúng ta có thể tính được độ dài của đường dựa theo hàm số bằng cách cộng các “đoạn”.



Ở mỗi đoạn, độ dài được xác định là độ dài đường thẳng của các đoạn thẳng rất nhỏ, cụ thể là

các đoạn thẳng từ M#, M # đến M#N , M #N hay K∆ + M #N − M # Và được “vi phân” hóa: MO # ∆ = M #N − M # . Và ta được:

K∆ + MO # ∆ = K1 + MO # ∆

Do đó, công thức tổng quát cho một đường cong khả vi từ P đến Q được xác định tổng các đoạn thẳng nhỏ, nghĩa là:

T V

T→U # ∆ = K1 + MO # ∆ lim 9 K1 + MO

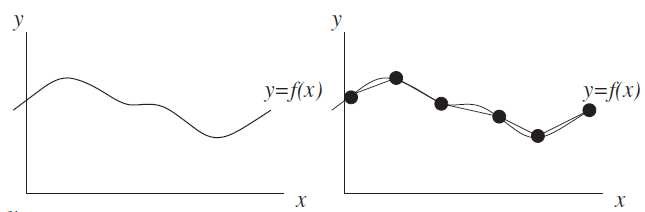
#? W

Tóm lại, độ dài của một đường cong trong đoạn [P, Q] là:

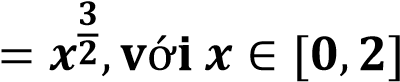
Z

XYZ = K[ + \O]^ \_∆‘

Y



**Thực hành 6: Hãy viết các câu lệnh Python để tìm độ dài của đường cong như sau:**

\ ‘ 

Gợi ý: Với 1 hàm f

* Hãy tính đạo hàm
* Sau đó đưa vào công thức tích phân tính đường cong fWV.

Thực hành:

* Khai báo gói symbol và khai biến, báo hàm f:

>>> ………… from sympy import Symbol, Derivative, Integral

…………..  sinh viên tự điền

>>> ………………… x = Symbol('x')

……………..  sinh viên tự điền

>>> ………………… f = x\*\*3/2

………………..  sinh viên tự điền

* Tính đạo hàm f:

>>> ………………… fx=Derivative(f, x).doit()

…………………..  sinh viên tự điền

* Tính tích phân theo đạo hàm của đường cong:

>>> ……………………… fx1=Integral(fx,(x,0,2))

……………..  sinh viên tự điền

Lưu ý: Mở rộng với một đường cong tham số theo , độ dài đường cong 3 chiều sẽ được tính toán theo công thức (nếu trên mặt phẳng thì g = 0):

Z

XYZ = Kh‘/h]\_ + hj/h] \_ + hk/h] \_∆]

Y

### 2.2. Bài toán “đường cong mềm mại”

**Cơ sở lý thuyết:**

Cho đường cong f, lấy 2 điểm I và I trên đường cong. Từ 2 điểm có các tiếp tuyến tương ứng là I và I . Góc giữa 2 tiếp tuyến là ∆l và độ dài cung I I là ∆m. Khi đó, ta gọi **độ cong** **trung bình** nVcủa đường f trên cung I I là tỷ số:

∆l nV ≔ p∆mp

**Độ cong** tại điểm I là giới hạn khi I tiến dần đến I và được kí hiệu là I

∆l l

I = qlim→q nV = ∆t→lim p∆mp = pmp r s

Người ta chứng minh được rằng với đường cong tham số bất kỳ thì:

up ′ J′p + pJ′ g′p + wg′ ′w

′′ J′′ g′′ g′′ g′′ ′′

= ′ + J′ + g′ x/

Ở đây đạo hàm đều lấy theo thông số .

**Bài toán thiết kế đường sắt**: ở đường sắt thường có các đoạn đường có độ cong khác nhau nối liền nhau. Không mất tính tổng quát, giả sử có một đoạn thẳng nối với một cung tròn tại điểm nối A. Khi đó, tại vị trí A, giả sử vận tốc của tàu là y không đổi thì thành phần gia tốc tiếp tuyến

z{ {}

= 0 nên toàn bộ gia tốc chỉ còn thành phần hướng tâm P = y | = |. Ta có trên đoạn zn ~ thẳng giá trị = 0 nên |P| = 0. Trên cung tròn, ta có |P| = {}.

Do đó, |P| xem như bị gián đoạn tại điểm A và tàu sẽ sinh ra một lực mạnh tại điểm A khi tàu đi qua (dễ gây hư hỏng tại điểm A). Để tránh hiệu ứng này, đường ray cần thiết kế sao cho **độ cong biến thiên liên tục**.

Lưu ý: Gia tốc P của một vật di chuyển trên đường cong sẽ là tổng của gia tốc tiếp tuyến và gia tốc pháp tuyến (hướng tâm) theo công thức như sau:

y

P = + y |

**Thực hành 7: Hãy vẽ 3 đoạn đường sắt sau khi thiết kế xong đoạn nối (**với−2 ≤ ≤ 2)

Yêu cầu đoạn nối: Giả sử có hai đoạn đường sắt: đoạn 1 ở phần < 0 của trục và đoạn 2 là = J, ≥ 1. Hãy thiết kế đoạn nối J = M , 0 ≤ ≤ 1 với M là đa thức bậc thấp nhất.

**Giải:**

Yêu cầu của bài này là chọn M để đường ray **liên tục**, có tiếp tuyến liên tục và có độ cong liên tục tại các vị trí = 0 và = 1. Áp dụng công thức về độ cong của đường là:

|MOO |

= }K 1 + MO x

Xét các giá trị M, MO, MOO tại = 0 và = 1 trên 2 đoạn thẳng (≤ 0 và ≥ 1), ta có:

M 0 = 0, MO 0 = 0, MOO 0 = 0,

M 1 = 1, MO 1 = 1, MOO 1 = 0

Từ 6 điều kiện độc lập trên, ta có thể xác định đa thức M là đa thức bậc 5 là đủ. Biểu thức f:

M = + + + x + +

M′ = + 2 + 3 + 4 x + 5

M′′ = 2 + 6 + 12 x + 20 x

* 3 điều kiện tại = 0 suy ra = = = 0.
* 3 điều kiện tại = 1 suy ra hệ phương trình:

+ + = M 1 = 1

3 + 4 + 5 = MO 1 = 1

6 + 12 + 20 = MOO 1 = 0

Giải hệ phương trình đại số tuyến tính này, ta được: 

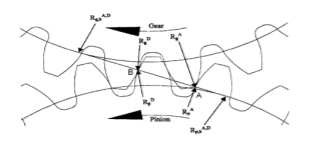
= = −8; = 3

Vậy: M = x + + =……… 2304\*x\*\*11…………..  sinh viên điền vào.

Từ đó, sử dụng Python để vẽ:

>>> ………………………………………………………..  Sinh viên thực hiện lệnh vẽ!

**Lời bàn:** bài toán “đường cong mềm mại” này thực chất có cơ sở toán là túc bế (evolute) và thân khai (involute). Do đó, các bạn Sinh viên có thể tìm đọc thêm trong các tài liệu để tìm hiểu rõ hơn hai khái niệm. Ngoài ứng dụng trên, túc bế và thân khai nghiên cứu ứng dụng trong các thiết kế cho các hệ cơ học như: bánh răng, hộp số nhằm giảm lực li tâm, lực ma sát gây ra hao mòn kim loại để vận hành êm ái và tăng độ bền cho các chi tiết trong cơ học.



**Lưu ý: Đối với đường cong tham số**

= ; J = J

Túc bế (evolute) được xác định bằng công thức:

O + JO

= + JO OOJO − JOO O

O O + JO

= J + JOO O − OOJO

# 4. Xử lý dãy số và chuỗi số trong thư viện sympy

Sympy hỗ trợ các lớp và thư viện hàm **series** để xử lý dãy số và chuỗi số. Cụ thể như sau:

## a. Dãy số trong sympy

**Thực hành 8: Xử lý dãy số (sequence) trong sympy**

 Sai phân cấp k của dãy số (mặc định k=1) với hàm difference\_delta

Ví dụ: ta có dãy số T = {- ∗ - + 1 }, nghĩa là số hạng thứ n là T = - + - ; TN =

* + 1 - + 2 = - + 3- + 2 và TN = - + 2 - + 3 = - + 5- + 6 . Do đó,
* Sai phân cấp 1 sẽ là: TN − T = 2 ∗ - + 2
* Sai phân cấp 2 sẽ là: TN − T = 4 ∗ - + 6

Thực hiện trên Sympy:

**>>> from sympy import difference\_delta as dd**

**>>> from sympy.abc import n**

>>> **dd**(n\*(n+1), n)

……………… 2\*n + 2……………………………… sinh viên ghi kết quả và giải thích

>>> **dd**(n\*(n+1), n, 2)

…………………… 4\*n + 6………………………… sinh viên ghi kết quả và giải thích

* Giới hạn dãy số: bằng hàm **limit\_seq** với hai tham số: hàm và giá trị biến chạy

**>>> from sympy.abc import n, k**

**>>> from sympy import limit\_seq**

>>> **limit\_seq**((3\*n\*\*3+5\*n\*\*2+4)/(5\*n\*\*3+200\*n\*\*2+400), n)

…………………………… 3/5………………………… sinh viên ghi kết quả và giải thích

>>> **limit\_seq**((3\*n\*\*3+5\*n\*\*2+4)/(5\*n\*\*3+200\*n\*\*2+400), n)

…………………………… 3/5……………………… sinh viên ghi kết quả và giải thích

Sinh viên hãy tính giới hạn:

T

2 1

T→Ulim 9 4 9 × T-

$ 2 ? ?

**>>> from sympy.abc import n, k, m**

>>> limit\_seq(Sum(k\*\*2 \* Sum(2\*\*m/m,(m,1,k)),(k,1,n))/(2\*\*n\*n), n)

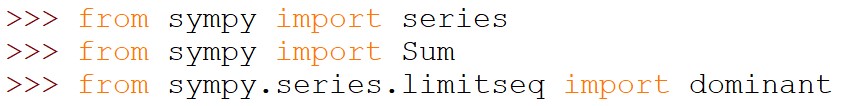
…………………………4…………………………… sinh viên ghi kết quả và giải thích

* Khái niệm hàm vô cùng lớn **dominant**: ví dụ - ≫ - khi - tiến về vô cùng. Khi đó, nếu trong biểu thức có 2 đại lượng l- + - thì đại lượng vô cùng lớn so với đại lượng còn lại chính là l- . Hàm **dominant** trong gói **sympy.series.limitseq**. Hàm **dominant** trả về None khi các biểu thức bằng nhau. Minh họa:

>>> **from** sympy import series

>>> **from** sympy import Sum

>>> **from** sympy.series.limitseq import dominant



>>> **from** sympy.abc import n,k

>>> dominant(n\*\*3+100\*n\*\*2+200\*n+1, n) n\*\*3

>>> dominant(2\*\*n + Sum(k, (k,0,n)), n)

……………………………2\*\*n……………………… sinh viên ghi kết quả và giải thích

>>> dominant(n\*\*2 + Sum(k, (k,0,n)), n)

…………………………None………………………… sinh viên ghi kết quả và giải thích

>>> dominant(2\*n\*\*2 + Sum(k, (k,0,n)), n)

……………………………None………………………… sinh viên ghi kết quả và giải thích

>>> dominant(n\*\*3 + Sum(k, (k,0,n)), n)

………………………n\*\*3……………………… sinh viên ghi kết quả và giải thích

>>> dominant(n\*3 + Sum(k, (k,0,n)), n)

……………………Sum(k, (k, 0, n))………………………… sinh viên ghi kết quả và giải thích

>>> dominant(2\*n\*3 + Sum(k, (k,0,n)), n)

………………………Sum(k, (k, 0, n))……………………… sinh viên ghi kết quả và giải thích

>>> dominant(2\*n\*\*3 + Sum(k, (k,0,n)), n)

……………………2\*n\*\*3…………………………… sinh viên ghi kết quả và giải thích

## b. Chuỗi số trong sympy

Sinh viên thực hành một số lệnh dưới đây về xử lý chuỗi số trong sympy:

**Thực hành 9: Xử lý chuỗi số (series) trong sympy**

* **Khai báo chuỗi số:**

Ví dụ: khai báo = ∑?T 4

>>> from sympy import Sum

>>> from sympy.abc import n, k

>>> S = Sum(k, (k, 0, n))

>>> S

………………………Sum(k, (k, 0, n))………………………….

* **Giới hạn chuỗi số:**

>>> from sympy import limit\_seq

>>> limit\_seq(S, n)

……………………oo……………………………… sinh viên ghi kết quả và giải thích.

Lưu ý: với hàm số, việc sử dụng hàm Limit của gói sympy.limit là tính toán giới hạn của hàm số (mà Sinh viên đã được học ở chương đầu tiên)

* **“Chuỗi hóa” một hàm số**

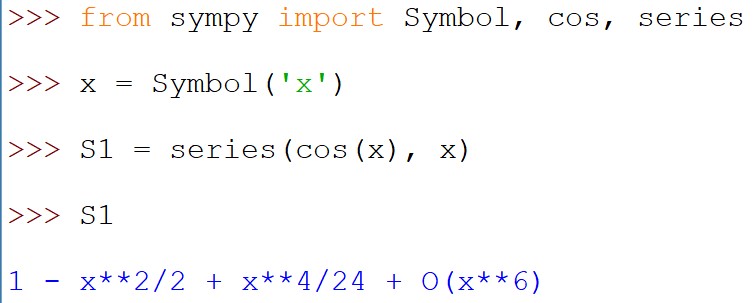
>>> from sympy import Symbol, cos, series

>>> x = Symbol('x')

>>> S1 = **series**(cos(x), x)

>>> S1

……………………2.7048138294………………………… sinh viên ghi kết quả và giải thích.



* **Tính toán chuỗi:**

>>> from sympy.abc import n

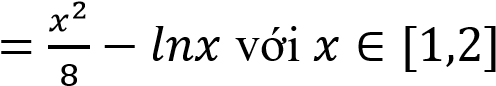
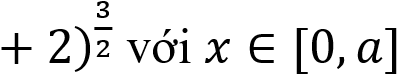
>>> e = (1+1/n)\*\*n

>>> print (round(e.subs(n, 100).evalf(),10))

………………………………………………………… sinh viên ghi kết quả và giải thích.

# BÀI TẬP CHƯƠNG 7

**Bài tập 1:** **Hãy viết các câu lệnh Python để tìm độ dài của các đường cong như sau**

1. M .
2. M = ¡ . x
3. M = ln m0- với ∈ [⁄ , ⁄]. x

**Bài tập 2:** **Tính giá trị F trong công thức CO với các thông số được cho như sau:**

& = 20 ¥.§n, 0 ≤ ≤ 10

Gợi ý: Để tính tích phân, đặt: , = , y = ¥.§n. Suy ra:

.§

, = ; y = ¥.§

, y = ,y − y ,

Giả định A là một giá trị 6.