Thuật toán tính gần đúng tích phân :

Chương trình chính  
Input:Output: In ra kết quả tính tích phân, sai số hoặc số khoảng chia cần thiết để đạt  
sai số cho trước  
Thuật toán  
B1: Khởi tạo một số biến toàn cục được sử dụng nhiều lần như:

f, a, b, n, h : Hàm số, khoảng lấy tích phân, số khoảng chia và bước h

|  |  |
| --- | --- |
| A | : Mảng chứa giá trị của hàm tại các mốc nội suy |
| D | : Mảng chứa hệ số của đa thức là tích các (𝑡 - 𝑖), i từ 0 đến n |

**B2**: Bước ℎ = (b-a)/n  
  
**B3:** Gọi ra các hàm tính toán các yêu cầu bài toán

Gói tính giá trị lớn nhất của đạo hàm cấp i của hàm |𝑓(𝑥)| trên đoạn [𝑎, 𝑏]  
Gói này tên là ***max  
Input:*** Hàm 𝑓 và i  
***Output:*** Giá trị lớn nhất của đạo hàm cấp i của hàm |𝑓(𝑥)| trên đoạn [𝑎, 𝑏]  
Thuật toán:  
**B1:** Hàm 𝑔(𝑥) = đạo hàm cấp i của 𝑓(𝑥)  
**B2:** Tìm 𝑚1, 𝑚2 lần lượt là giá trị nhỏ nhất và giá trị lớn nhất của hàm  
𝑔(𝑥) trên đoạn [𝑎, 𝑏] bằng thư viện có sẵn  
**B3:** So sánh |𝑚1| và |𝑚2|, khi đó giá trị lớn nhất của |𝑓(𝑥)| là giá trị lớn hơn  
trong |𝑚1| và |𝑚2|

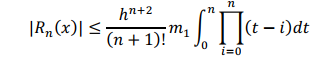
Gói nhân một đa thức với (𝑡 - 𝑖) ***Input:*** Mảng A chứa hệ số của đa thức cần nhân, ví dụ mảng A của đa thức  
𝑥2 - 3𝑥 + 5 là [1, -3, 5] và i  
***Output:*** Mảng A chứa chính hệ số sau khi nhân với i  
Thuật toán:  
**B1:** Thêm 0 vào cuối mảng A  
**B2:** m = độ dài mảng A  
**B3:** Với j từ m-1 chạy về 1  
A[j] = A[j] - A[j - 1] \* i  
**B4:** Trả về mảng A

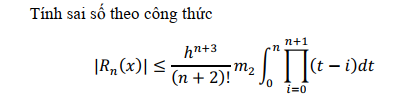
Gói chia một đa thức với (𝑡 - 𝑖)(chia hết)

***Input:*** Mảng A chứa hệ số của đa thức cần chia, ví dụ mảng A của đa thức  
𝑥2 - 3𝑥 + 5 là [1, -3, 5] và i  
***Output:*** Mảng X chứa hệ số của đa thức chia  
Thuật toán:  
**B1:** Với j chạy từ 1 đến độ dài của X  
X[j] = i\*X[j-1] + A[j]  
**B2:** Trả về X  
Gói tính tích phân xác định của một đa thức  
***Input:*** Mảng A chứa hệ số của đa thức, a, b là khoảng lấy tích phân  
***Output:*** Giá trị tích phân xác định  
Thuật toán  
**B1:** I = 0  
**B2:** m = độ dài mảng A  
**B3:**Với j chạy từ 0 đến m-1  
Nếu A[j] = 0  
Bỏ qua  
Nếu A[j] ≠ 0  
A[j] = A[j]/(m-j)  
I = I + A[j]\*(b^(m-j))-a^(m-j))  
**B4:** Trả về I  
Gói tính hệ số Cotes 𝐻𝑖  
***Input:*** Hệ số Cotes thứ i  
***Output:*** Hệ số Cotes 𝐻𝑖  
Thuật toán:  
**B1:** Tạo mảng X chứa hệ số của phép chia đa thức D (tích các (t-i), i từ 0 đến  
n) với (t-im  
**B2:** Tính hệ số 𝐻𝑖 theo công thức

𝐻𝑖 =  
1 𝑛  
(-1)𝑛-𝑖  
𝑖! (𝑛 - 𝑖)! ∫0𝑛 ∏𝑛 𝑗=𝑡0(-𝑡 𝑖- 𝑗) 𝑑𝑡  
**B3:** Trả về 𝐻𝑖  
Gói in ra các hệ số Cotes và tính giá trị của tích phân  
(𝑏 - 𝑎) ∑ 𝐻𝑖𝑦𝑖  
𝑛  
𝑖=0  
***Input:  
Output:*** Kết quả tích phân  
Thuật toán:  
**B1:** Khởi tạo biến I để tính tích phân, tạo mảng H để chứa các hệ số 𝐻𝑖  
**B2:** Với i chạy từ 0 đến n  
𝐻𝑖 = hệ số Cotes thứ i  
𝐼 = 𝐼 + 𝐻𝑖 ∗ 𝑦𝑖  
**B3:** In kết quả I\*(b-a)

Gói tính sai số của công thức Newton – Cotes ***Input:  
Output:*** Sai số công thức Newton - Cotes  
Thuật toán  
**B1:** Tạo hàm 𝑔 là đạo hàm cấp n của hàm 𝑓  
**B2:** Nếu n lẻ  
𝑚1 = GTLN của |𝑔′(𝑥)| trên đoạn [𝑎, 𝑏] bằng hàm ***max***Tính sai số theo công thức:

  
**B3:** Nếu m chẵn  
Tìm 𝑚2 = GTLN của |𝑔′′(𝑥)| trên đoạn [a, b] bằng hàm ***max***

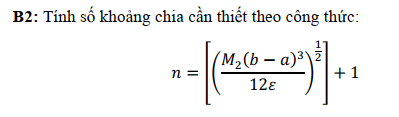


**B4:** In ra giá trị sai số

Gói tính tích phân bằng công thức hình thang  
***Input:*** Mảng A chứa giá trị của hàm 𝑓(𝑥) tại các mốc 𝑥𝑖  
***Output:*** Giá trị gần đúng của tích phân xác định bằng công thức hình  
Thuật toán  
**B1: **

**B2:** Với i trong khoảng (1, n)  
𝑡𝑟𝑎𝑝𝑒 = 𝑡𝑟𝑎𝑝𝑒 + 𝐴[𝑖]  
**B3:** In 𝑡𝑟𝑎𝑝𝑒  
Gói tính sai số của tích phân bằng công thức hình thang  
***Input:  
Output:*** Sai số của tích phân bằng công thức hình thang  
Thuật toán  
**B1:** Tìm 𝑀2 = max 𝑥∈[𝑎,𝑏] |𝑓′′(𝑥)| bằng hàm ***max*B2:** Sai số tính theo công thức 

Gói tính số khoảng chia cần thiết để đạt sai số 𝜖 cho trước  
***Input:  
Output:*** Số khoảng chia cần thiết  
Thuật toán  
**B1:** Tìm 𝑀2 = max 𝑥∈[𝑎,𝑏] |𝑓′′(𝑥)| bằng hàm ***max***



Gói tính gần đúng tích phân bằng công thức Simpson ***Input:*** Mảng A chứa giá trị của hàm 𝑓(𝑥) tại các mốc 𝑥𝑖  
***Output:*** Giá trị gần đúng của tích phân xác định bằng công thức Simpson  
Thuật toán  
**B1:** simp = h/3\*(A[0]+A[n])  
simp\_odd = 0  
simp\_even = 0  
**B2:** Với i từ 1 đến n, i lẻ  
simp\_odd += A[i]  
**B3:** Với i từ 2 đến n, i chẵn  
simp\_even += A[i]  
**B4:** simp = simp + h/3\*4\*simp\_odd + h/3\*2\*simp\_even  
Gói tính sai số công thức tính gần đúng tích phân bằng công thức Simpson  
***Input:  
Output:*** Sai số công thức tính gần đúng tích phân bằng công thức Simpson  
Thuật toán  
**B1:** Tìm 𝑀2 = max  
𝑥∈[𝑎,𝑏]  
|𝑓(4)(𝑥)| bằng hàm ***max*B2:** Tính sai số theo công thức



Gói tính số khoảng chia cần thiết để đạt sai số 𝜖 cho trước  
***Input:***

***Output:*** Số khoảng chia cần thiết  
Thuật toán  
**B1:** Tìm 𝑀2 = max  
𝑥∈[𝑎,𝑏]  
|𝑓(4)(𝑥)| bằng hàm ***max*B2:** Tính số khoảng chia cần thiết theo công thức:

s