Nguyễn Trần Ánh Linh-47.01.104.123

**Mô tả bài toán:**

* ***Đầu vào:***
* Bậc của đa thức (bac): Người dùng nhập giá trị bậc của đa thức.
* Các hệ số của đa thức (heSo): Người dùng nhập các hệ số của đa thức theo thứ tự tăng dần theo bậc.
* Điểm x để tính giá trị đa thức: Người dùng nhập giá trị x.
* ***Đầu ra:***
* Kết quả là giá trị của đa thức tại điểm x (được lưu trong biến ketQua).
* ***Cách thực thi:***
* Khai báo và khởi tạo các biến và vector heSo.
* Yêu cầu người dùng nhập bậc của đa thức (bac).
* Khởi tạo vector heSo với kích thước (bac + 1).
* Yêu cầu người dùng nhập các hệ số của đa thức theo thứ tự tăng dần theo bậc và lưu trữ vào heSo.
* Yêu cầu người dùng nhập điểm x để tính giá trị đa thức.
* Gọi hàm horner(heSo, x) để tính giá trị đa thức tại điểm x.
* Kết quả tính toán được lưu trữ trong biến ketQua.
* In kết quả giá trị đa thức tại điểm x ra màn hình.

**Source Code:**

#include <iostream>

#include <vector>

using namespace std;

double horner(const vector<double>& heSo, double x) {

double ketQua = 0;

for (vector<double>::const\_reverse\_iterator it = heSo.rbegin(); it != heSo.rend(); ++it) {

ketQua = ketQua \* x + \*it;

}

return ketQua;

}

int main() {

int bac; // Bậc của đa thức

cout << "Nhap bac cua da thuc: ";

cin >> bac;

vector<double> heSo(bac + 1); // Khởi tạo vector heSo với kích thước bằng (bac + 1)

cout << "Nhap cac he so cua da thuc (tang dan theo bac):\n";

for (int i = bac; i >= 0; --i) {

cout << "Nhap he so thu " << i << ": ";

cin >> heSo[i];

}

double x; // Điểm x để tính giá trị đa thức

cout << "Nhap diem x: ";

cin >> x;

double ketQua = horner(heSo, x);

cout << "Gia tri cua da thuc tai diem x = " << x << " la: " << ketQua << endl;

return 0;

}

**Giải thích source code:**

* Hàm horner là nơi thực hiện thuật toán Horner để tính giá trị của đa thức. Hàm này nhận đầu vào là vector heSo chứa các hệ số của đa thức theo thứ tự tăng dần theo bậc và giá trị x để tính giá trị đa thức tại điểm đó.

(Giải thích chi tiết hàm :

// Hàm tính giá trị của đa thức sử dụng thuật toán Horner

Đây là một hàm có kiểu trả về là double.

+ Hàm nhận hai đối số: heSo là vector chứa các hệ số của đa thức, và x là điểm để tính giá trị đa thức tại đó.

double ketQua = 0;

+ Khởi tạo biến ketQua với giá trị ban đầu là 0. Biến này sẽ lưu trữ giá trị của đa thức khi tính toán.

+ for (vector<double>::const\_reverse\_iterator it = heSo.rbegin(); it != heSo.rend(); ++it)

Bắt đầu vòng lặp để lặp qua các hệ số của đa thức.

Sử dụng const\_reverse\_iterator để lặp ngược từ hệ số cao nhất đến hệ số thấp nhất.

+ Vòng lặp sẽ dừng khi đã lặp qua tất cả các hệ số của đa thức.

ketQua = ketQua \* x + \*it;

+ Trong mỗi lần lặp, tính toán giá trị của đa thức tại điểm x theo thuật toán Horner.

Biểu thức này nhân giá trị hiện tại của ketQua với x và cộng với hệ số hiện tại được trỏ bởi it.

return ketQua;

+Trả về giá trị của đa thức tại điểm x sau khi đã lặp qua tất cả các hệ số.)

* Trong hàm, một biến ketQua được khởi tạo ban đầu với giá trị 0. Sau đó, một vòng lặp for được sử dụng để lặp qua các hệ số của đa thức từ hệ số cao nhất đến thấp nhất. Với mỗi hệ số, giá trị ketQua được tính toán bằng cách nhân với x và cộng với hệ số hiện tại.
* Trong hàm main, chương trình bắt đầu bằng việc yêu cầu người dùng nhập vào bậc của đa thức thông qua biến bac. Tiếp theo, một vector heSo được khởi tạo với kích thước (bac + 1) để chứa các hệ số của đa thức.
* Vòng lặp for được sử dụng để yêu cầu người dùng nhập vào các hệ số của đa thức theo thứ tự tăng dần theo bậc. Hệ số thứ i sẽ được lưu trữ trong heSo[i].
* Tiếp theo, người dùng được yêu cầu nhập giá trị x để tính giá trị của đa thức tại điểm đó.
* Sau khi có đầy đủ thông tin, hàm horner được gọi với vector heSo và giá trị x, và kết quả được lưu trữ trong biến ketQua.
* Cuối cùng, kết quả của đa thức tại điểm x được in ra màn hình thông qua câu lệnh cout.