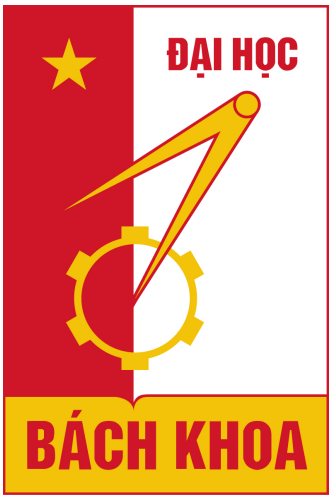
ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

o0o



BÁO CÁO BÀI TẬP LỚN

MÔN HỌC: KỸ THUẬT LẬP TRÌNH

CHỦ ĐỀ 3:

GIẢI GẦN ĐÚNG PHƯƠNG TRÌNH $f(x)$ BẰNG PHƯƠNG PHÁP DÂY CUNG

Giảng viên hướng dẫn: TS.Nguyễn Thị Thanh Huyền

Sinh viên thực hiện: Nguyễn Văn Tuấn - 20216965

Mã lớp học: 142298

Hà Nội, 8/2023.

Mục lục

1 Bài toán

1.1 Đề bài:

Chủ đề 3: Viết chương trình giải gần đúng phương trình $f(x) = 0$, trong đó $f(x)$ là đa thức, bằng phương pháp dây cung. Thực hiện các yêu cầu sau:

1. Tìm các miền chứa nghiệm của phương trình.
2. Tìm khoảng phân ly nghiệm (a, b) của phương trình, thoả mãn $|a - b| \leq 0.5$, bằng cách sử dụng phương pháp chia đôi để thu hẹp dần một khoảng phân ly nghiệm đã tìm được ở ý 1).
3. Tìm nghiệm gần đúng với số lần lặp n cho trước trong khoảng phân ly nghiệm (a, b) và đánh giá sai số theo cả hai công thức (n được nhập vào từ bàn phím, (a, b) có thể lấy từ kết quả của ý 2) hoặc được nhập vào từ bàn phím.
4. Tìm nghiệm gần đúng trong khoảng (a, b) với sai số e cho trước (e được nhập vào từ bàn phím, (a, b) có thể lấy từ kết quả của ý 2) hoặc được nhập vào từ bàn phím. Tính toán theo 2 cách áp dụng công thức sai số.
5. Tìm nghiệm gần đúng x_n trong khoảng (a, b) thoả mãn điều kiện: $|x_n - x_{n-1}| \leq e$ (e được nhập vào từ bàn phím).

Yêu cầu:

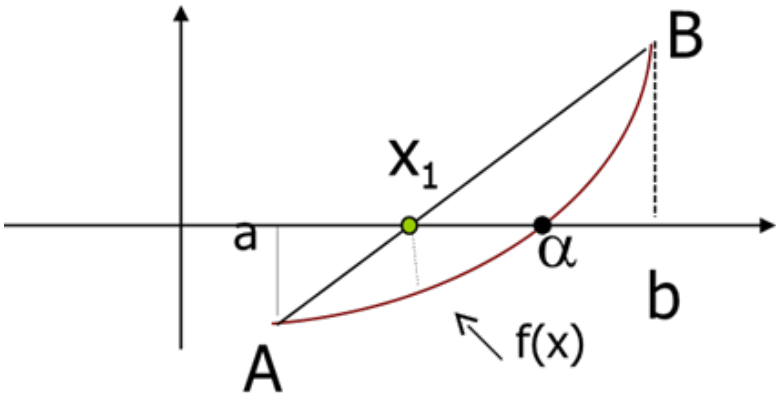
- Mọi kết quả được hiển thị với số chữ số phần thập phân nhập vào từ bàn phím.
- Ghi vào tệp văn bản thể hiện quá trình thực hiện chương trình và các kết quả ra.
- Thực hiện chương trình bằng menu điều khiển bởi các phím chức năng. Sinh viên tự code để thiết lập và điều khiển menu.

1.2 Phương pháp dây cung

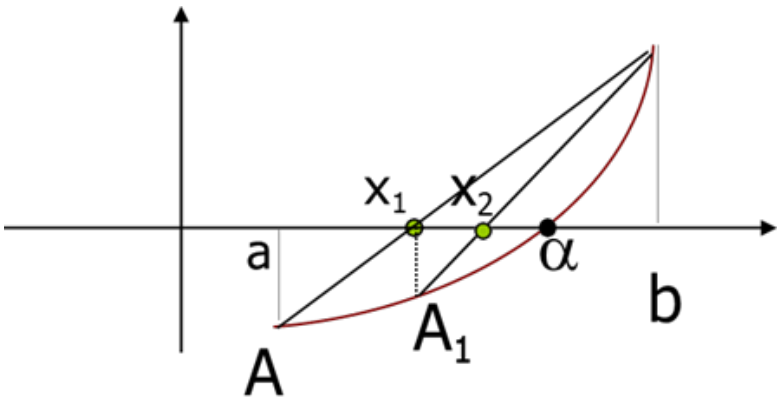
Bài toán: Giả sử (a, b) là khoảng phân ly nghiệm của phương trình $f(x) = 0$. Ta muốn tìm nghiệm thực gần đúng của $f(x) = 0$ trên (a, b) với sai số e cho trước.

1.2.1 Nội dung của phương pháp:

- Thay cung AB bằng dây trương cung AB. Dây trương cung AB cắt trục hoành tại điểm $(x_1, 0)$.



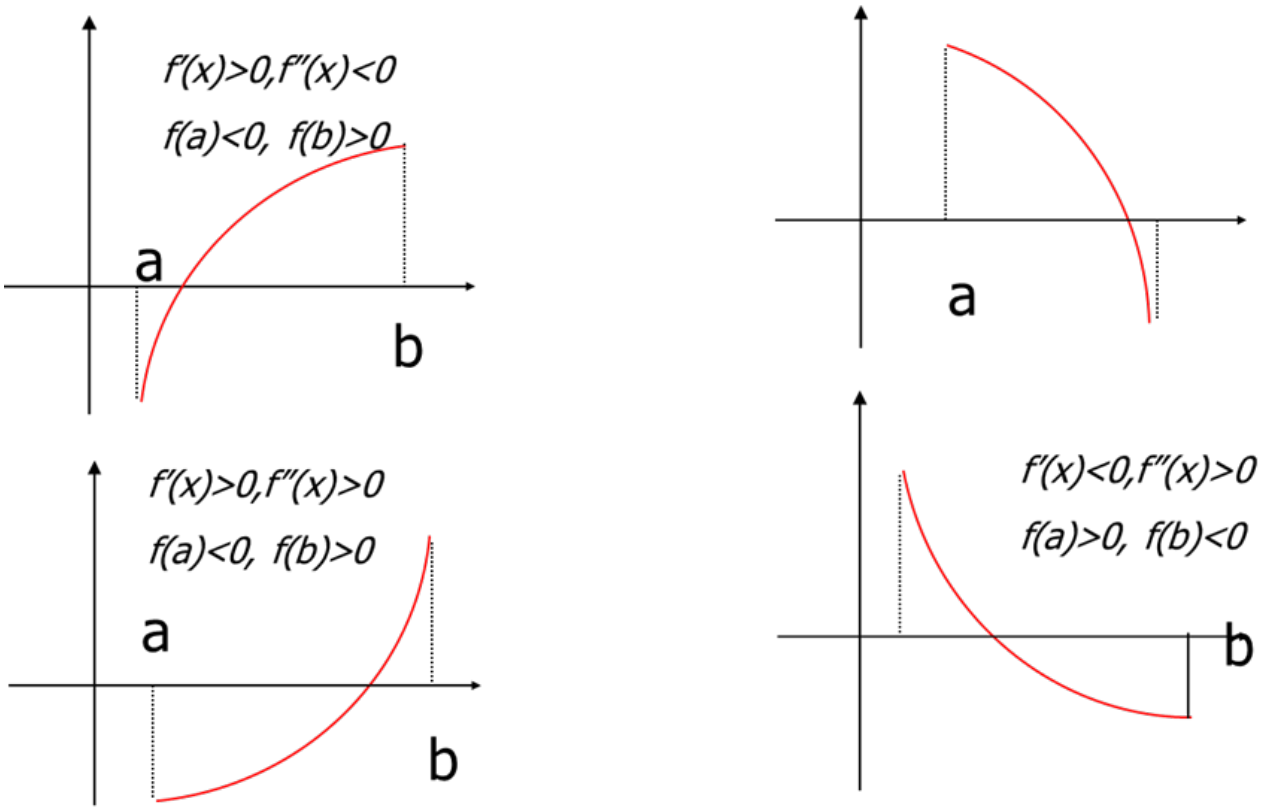
- Nếu $|x_1 - a| \leq e$, thì x_1 là nghiệm gần đúng cần tìm.
- Nếu không, lặp lại phương pháp dây cung với khoảng phân ly mới (x_1, b) hoặc (a, x_1) tùy theo tính chất của $f(x)$:
  - Nếu $f(x_1) \cdot f(a) < 0$, thì (a, x_1) là khoảng phân ly nghiệm mới.
  - Nếu $f(x_1) \cdot f(a) > 0$, thì (x_1, b) là khoảng phân ly nghiệm mới.



- Với khoảng phân ly nghiệm mới (x_1, b), tính được nghiệm gần đúng x_2 bằng phương pháp dây cung.
- Quá trình lặp kết thúc khi tìm được nghiệm gần đúng x_n có sai số $|x_n - x_{n-1}| \leq e$.

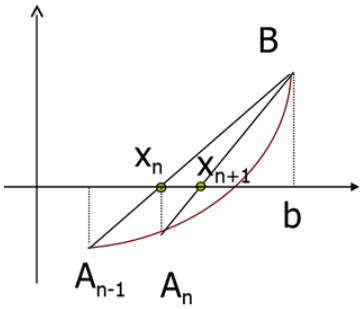
1.2.2 Công thức tính nghiệm

Để xây dựng công thức tính nghiệm, ta xét tính tăng giảm và tính lồi lõm của đường cong $f(x)$. Giả sử f' và f'' không đổi dấu trên (a, b).



Trường hợp: $f'(x)f''(x) > 0$

- Chọn $x_0 = a$
- Ở bước thứ n, phương trình đường thẳng AB_n là:



$$\frac{y - f(x_n)}{f(b) - f(x_n)} = \frac{x - x_n}{b - x_n}$$

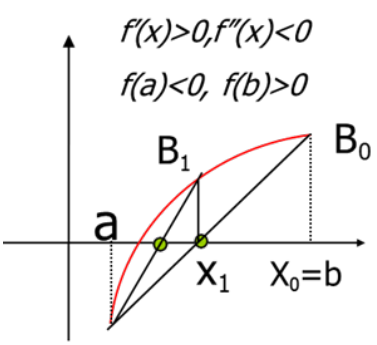
X_{n+1} là nghiệm của hệ:

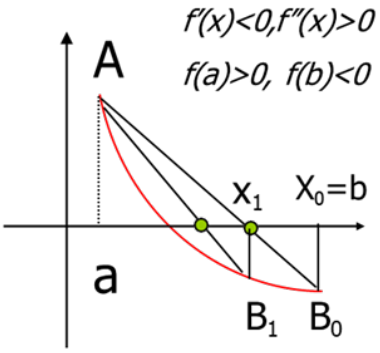
$$\begin{cases} \dfrac{y - f(x_n)}{f(b) - f(x_n)} = \dfrac{x - x_n}{b - x_n} \\ y = 0 \end{cases}$$

$$x_{n+1} = x_n - \frac{f(x_n).(x_n - b)}{f(x_n) - f(b)}$$

$$x_0 = a$$

Trường hợp: $f'(x)f''(x) < 0$





- Chọn $x_0 = b$
- Phương trình đường thẳng AB_0:

$$\frac{y - f(x_0)}{f(a) - f(x_0)} = \frac{x - x_0}{a - x_0}$$

Ở bước n, phương trình đường thẳng AB:

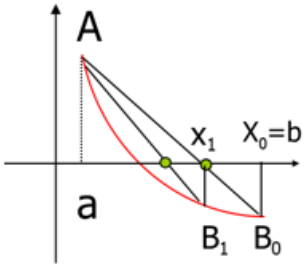
$\mathbf{x_1}$: là nghiệm của hệ: $\begin{cases} \frac{y - f(x_0)}{f(a) - f(x_0)} = \frac{x - x_0}{a - x_0} \\ y = 0 \end{cases} \Rightarrow x_1 = x_0 - \frac{f(x_0)}{f(x_0) - f(a)}(x_0 - a)$

Nghiệm gần đúng X_{n+1} cần tìm là nghiệm của hệ:

$$\begin{cases} \frac{y - f(x_n)}{f(a) - f(x_n)} = \frac{x - x_n}{a - x_n} \\ y = 0 \end{cases}$$

$$x_{n+1} = x_n - \frac{f(x_n).(x_n - a)}{f(x_n) - f(a)}$$

Với $X_0 = b$



Từ hai trường hợp trên, ta rút ra công thức tính nghiệm chung:

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - d)}{f(x_n) - f(d)}$$

Trong đó:

- Nếu $d = b$ và $x_0 = a$ nếu $F(b)$ cùng dấu với $F''(x)$ (hay $F'(x)F''(x) > 0$).
- Nếu $d = a$ và $x_0 = b$ nếu $F(a)$ cùng dấu với $F''(x)$ (hay $F'(x)F''(x) < 0$).

Lưu ý: Ở phương pháp này ta không xét trường hợp $F'(x)F''(x) = 0$ vì nếu bằng 0 thì một trong hai giá trị $F'(x) = 0$ hoặc $F''(x) = 0$.

- Nếu $F''(x) = 0$ thì $F'(x)$ là hàm hằng nên $F(x)$ là hàm bậc nhất (là một đường thẳng), do đó không thể xác định tính lồi lõm của hàm.
- Nếu $F'(x) = 0$ thì ta sẽ không tìm được $F''(x)$ và $F(x)$ sẽ là hàm hằng, nên không xét được tính lồi lõm của hàm.

1.2.3 Đánh giá sai số

Sau khi xây dựng công thức $x_n = x_{n-1} - \frac{f(d)-f(x_{n-1})}{f(d)-f(x_{n-1})} \cdot f(x_{n-1})$ (*), ta cần chứng minh rằng nó hội tụ tới nghiệm của phương trình $f(x) = 0$.

Điều kiện hội tụ

- (a, b) là khoảng cách ly nghiệm.
- f', f'' liên tục và f'' không đổi dấu trên $[a, b]$.
- Chọn đúng điểm x_0, d.

Các bước chứng minh

- Dãy x_n đơn điệu và bị chặn.
- Giới hạn của dãy là nghiệm của phương trình.
- Chứng minh các công thức sai số.

Do f' và f'' không đổi dấu nên ta có 4 trường hợp:

- $f' > 0$ và $f'' > 0$
- $f' > 0$ và $f'' < 0$
- $f' < 0$ và $f'' > 0$
- $f' < 0$ và $f'' < 0$

CHỨNG MINH DÃY ĐƠN ĐIỆU VÀ BỊ CHẶN

Xét trường hợp $f' < 0$, $f'' > 0$. Ta có $f(a) > 0$ và $f(b) < 0$, suy ra $d = a$ và $x_0 = b$ (Do điểm Fourier $f(d) \cdot f''(d) > 0$).

- Giả sử (a, b) là khoảng ly nghiệm, f', f'' liên tục, $f'' > 0$ và f'' không đổi dấu trong (a, b). Khi đó, $f(x)$ là hàm lồi và dây cung MM_0 luôn nằm trên đồ thị.

Ta thấy $M(d, f(d))$ và $M_0(x_0, f(x_0))$ nằm ở hai phía của trục hoành. Dây cung MM_0 cắt trục hoành tại điểm $(x_1, 0)$.

Gọi $y(x)$ là hàm với đồ thị là đường thẳng MM_0. Dây cung MM_0 luôn nằm trên đồ thị $f(x)$, tức là $y(x_1) = 0 > f(x_1)$

Lại có: $y(x_0) < y(x_1) < y(d)$, suy ra $x_0 > x_1 > d$ (do $f' < 0$ nên f là hàm giảm trên $[a, b]$).

Suy ra, từ $f(x_0) < 0$, ta chứng minh được $x_0 > x_1 > d$ và $f(x_1) < 0$. Chứng minh tương tự, ta được $f(x_k) < 0$ và $x_{k-1} > x_k > d$ với mọi x.

Dãy x_n đơn điệu giảm và bị chặn. Do đó, dãy x_n hội tụ đến giá trị x^*, từ (*) khi $n \to \infty$ ta có:

$$x^* = x^* - \frac{f(x^*)(x^* - d)}{f(x^*) - f(d)}$$

Từ đây suy ra $f(x^*) = 0$, hay x^* chính là nghiệm của phương trình $f(x) = 0$.

Chứng minh công thức sai số

$m_1 = \min |f'(x)|$ và $M_1 = \max |f'(x)|$ với $x \in [a, b]$.

Công thức sai số mục tiêu:

$$|x_n - x^*| \leq \frac{|f(x_n)|}{m_1}$$

Công thức sai số theo hai xấp xỉ liên tiếp x_{n-1}:

$$x_n = x_{n-1} - \frac{(f(x_{n-1})(x_{n-1} - d))}{(f(x_{n-1}) - f(d))}$$

Áp dụng định lý Lagrange ta có:

$$f'(c_1)(d - x_{n-1})(x_n - x_{n-1}) = f'(c_2)(x^* - x_{n-1})(d - x_{n-1})$$

Từ đó suy ra:

$$f'(c_1)(x_n - x_{n-1}) = f'(c_2)((x^* - x_n) + (x_n - x_{n-1}))$$

Tương đương với:

$$(x^* - x_n) = \frac{(f'(c_1) - f'(c_2))}{f'(c_2)}(x_n - x_{n-1})$$

$|x^* - x_n| = \left|\frac{(f'(c_1)-f'(c_2))}{f'(c_2)}\right| |x_n - x_{n-1}| \leq \left|\frac{M_1-m_1}{m_1}\right| |x_n - x_{n-1}|$

Hay $|x^* - x_n| \leq \frac{M_1-m_1}{m_1}|x_n - x_{n-1}|$

1.3 Tìm miền chứa nghiệm

Định lý 1: Xét phương trình

$$p(x) = a_0x^n + a_1x^{n-1} + \ldots + a_{n-1}x + a_n = 0 \quad (a_0 \neq 0, a_i \in \mathbb{R}, \text{với } i = 1, n) \quad (1)$$

Đặt $A = \max\{|a_i|, i = 1, n\}$ thì các nghiệm (thực hoặc phức) của phương trình (1) thỏa mãn

$$|x| < 1 + \frac{A}{|a_0|} \quad (2)$$

Nghĩa là các nghiệm của phương trình (1) nằm trong mặt tròn tâm O bán kính $R = 1 + \frac{A}{|a_0|}$ ở trong mặt phẳng phức.

Chứng minh định lý: Xét các số x không thỏa mãn (2), tức là

$$|x| \geq 1 + \frac{A}{|a_0|} \ (3).$$

Từ đó suy ra $|a_0| \geq \frac{A}{|x|-1}$ hoặc $|a_0x^n| \geq \frac{A|x^n|}{|x|-1}$. Mặt khác

$$|a_1x^{n-1} + a_2x^{n-2} + \ldots + a_n| \leq |a_1||x^{n-1}| + \ldots + |a_n| \leq A(|x|^{n-1} + |x|^{n-2} + \ldots + 1) = \frac{A|x^{n-1}|}{|x| - 1}$$

Do (3) thì $|x| > 1$ nên $\frac{|x^{n-1}|}{|x|-1} < \frac{|x^n|}{|x|-1} \Rightarrow |a_1x^{n-1} + a_2x^{n-2} + \ldots + a_n| < |a_0x^n|$. Ta thấy số hạng đầu trội hơn hẳn các số hạng sau của đa thức. Do đó x thỏa mãn (3) thì không thể là nghiệm của phương trình (1). Định lý được chứng minh.

Từ đây ta có hệ quả: Số $N = 1 + \frac{A}{|a_0|}$ là cận trên của modun các nghiệm của phương trình (1).

Định lý 2: Nếu $a_0 > 0$, $a_i > 0$ với $i = 1, n$ thì $p(x) > 0 \forall x > 0$; phương trình (1) không có nghiệm dương.
Giả sử $a_0 > 0$, $a_k \geq 0$ ($k \geq 1$) là hệ số âm đầu tiên tính từ $k = 1, 2, ...$ gọi $B =$ max của các trị tuyệt đối của các hệ số âm. Thì các nghiệm dương của phương trình (1) thỏa mãn

$$x < 1 + \sqrt[k]{\frac{B}{a_0}}$$

Cách chứng minh hoàn toàn tương tự chứng minh định lý 1.

Ví dụ: Cho phương trình $p(x) = x^4 - 4x^3 - 7x^2 - 5x + 3$ (*)

1. Tìm miền chứa nghiệm (thực và phức)

2. Tìm cận trên, cận dưới miền chứa nghiệm thực

Giải:

- Miền chứa nghiệm: $a_0 = 1$, $A = \max\{1, 4, 7, 5, 3\} = 7$
  Vậy $N = 1 + \frac{A}{a_0} = 1 + \frac{7}{1} = 8$
  $\rightarrow$ Miền chứa nghiệm là miền tròn tâm O bán kính $R = 7$ nằm trong mặt phẳng phức.

- Tìm cận trên, dưới của nghiệm thực.
  Cận trên: hệ số âm đầu tiên là $a_1 = -4$ do đó $k = 1$, còn $B = \max\{|-4|, |-7|, |-5|\} = 7$
  $\rightarrow$ Cận trên của nghiệm thực dương là

$$x < M = 1 + \sqrt[1]{\frac{7}{1}} = 8$$

  Ta tìm cận dưới của các nghiệm thực âm. Từ phương trình (*) ta thay x bởi $-x$:

$$p(-x) = x^4 + 4x^3 - 7x^2 + 5x + 3 = 0$$

  Hệ số âm đầu tiên $a_2 = -7 \rightarrow k = 2$ và $B = |-7| = 7$

$$=> -x < M = 1 + \sqrt{\frac{7}{1}} = 3.645 \text{ hay } x > -M = -3.645$$

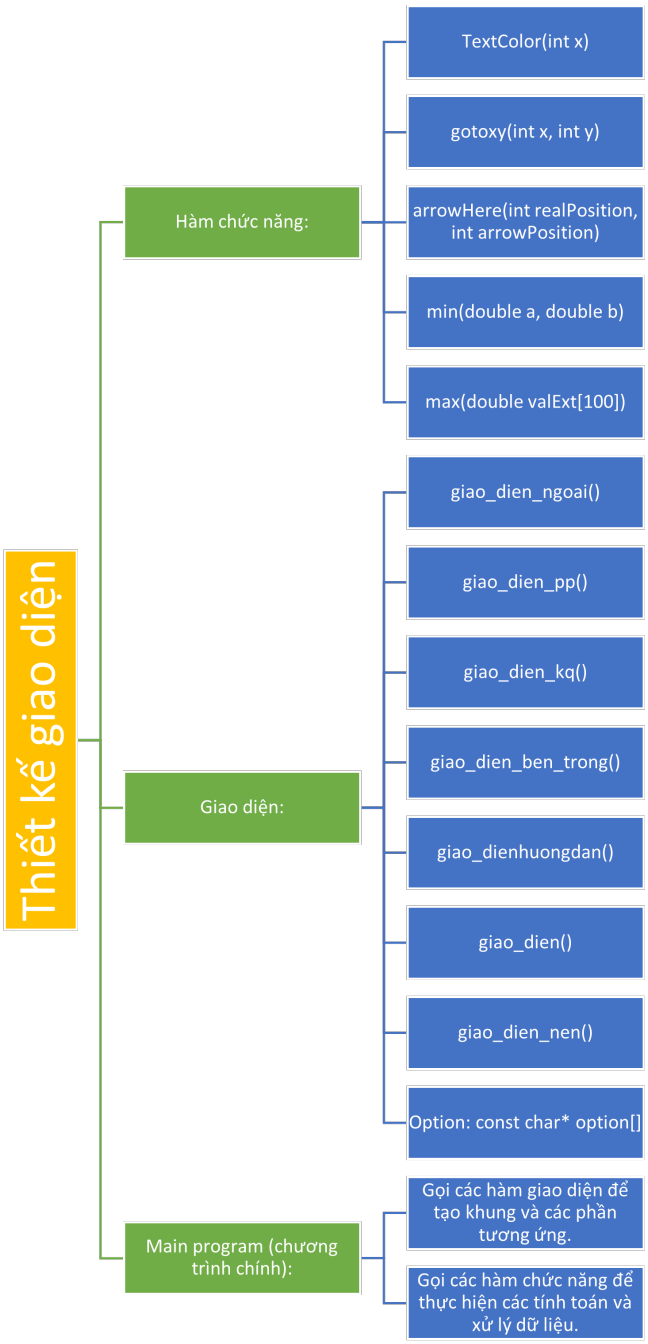
  Vậy nghiệm thực của phương trình (*) nằm trong khoảng $-3.645 < x < 8$

2 Quá trình thiết kế chương trình

Quá trình thiết kế chương trình gồm các 2 phần chính đó là:

- Thiết kế giao diện.
- Viết mã nguồn để chạy được các yêu cầu của chủ đề 3.

2.1 Hàm thiết kế giao diện



Để thiết kế giao diện em đã khai báo:

```
    void gotoxy(int x, int y)
{
    COORD c = { x, y };
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), c);
}
```

Hàm này để di chuyển con trỏ tới địa điểm với toạ độ x,y trên màn hình.
Tọa độ trên màn hình Console thì hơi ngược một chút với tọa độ (x,y) trong toán học. Gốc O sẽ từ vị trí bên trái và phía trên màn hình, tức là trục y sẽ ngược hướng xuống dưới.
Tiếp theo là khai báo:

```
    void TextColor(int x)
{
    HANDLE h = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(h, x);
}
```

Hàm này để đổi màu văn bản, với x là màu cần truyền vào. Tiêp theo là khai báo:

```
    void arrowHere(int realPosition, int arrowPosition)
{
    if (realPosition == arrowPosition)
    {
        TextColor(236); printf("%c  ", 175);
    }
    else {
        TextColor(252);  printf("    ");
    }
}
```

Giúp người dùng dễ dàng nhận biết vị trí hiện tại hoặc vị trí đang được chọn trong một giao diện dòng lệnh. Mũi tên có thể hỗ trợ người dùng trong việc di chuyển, lựa chọn hoặc làm rõ trạng thái hiện tại của chương trình hoặc ứng dụng.
Đây là 3 phần khai báo chính trong toàn bộ giao diện của chương trình. Tiếp theo ta viết các hàm giao_dien_ngoai(), giao_dien_trong(), giao_dien_kq(), giao_dien_huongdan(), giao_dien_pp(), giao_dien(), giao_dien_nen(). Các giao diện này dình các câu lệnh gotoxy(x,y), TextColor(x) và các vòng lặp for để vẽ các viền xung quanh hoặc đê đổ màu cho một khoảng được khai báo.Đây là ví dụ minh hoạ, cụ thể là giao_dien_kq():

```
    void giao_dien_kq()
{
  TextColor(236);
  for(int i=71; i<=118;i++){
  TextColor(236); for(int j=1;j<29;j++){
      gotoxy(i,j);
      printf(" ");
    }
  }
    TextColor(15);
    for (int i = 71; i <= 118; i++)
    {
        gotoxy(i, 0); printf("%c", 205);
        gotoxy(i, 29); printf("%c", 205);
    }

    gotoxy(0, 0); printf("%c",  201);
    gotoxy(118, 0); printf("%c", 187);
```

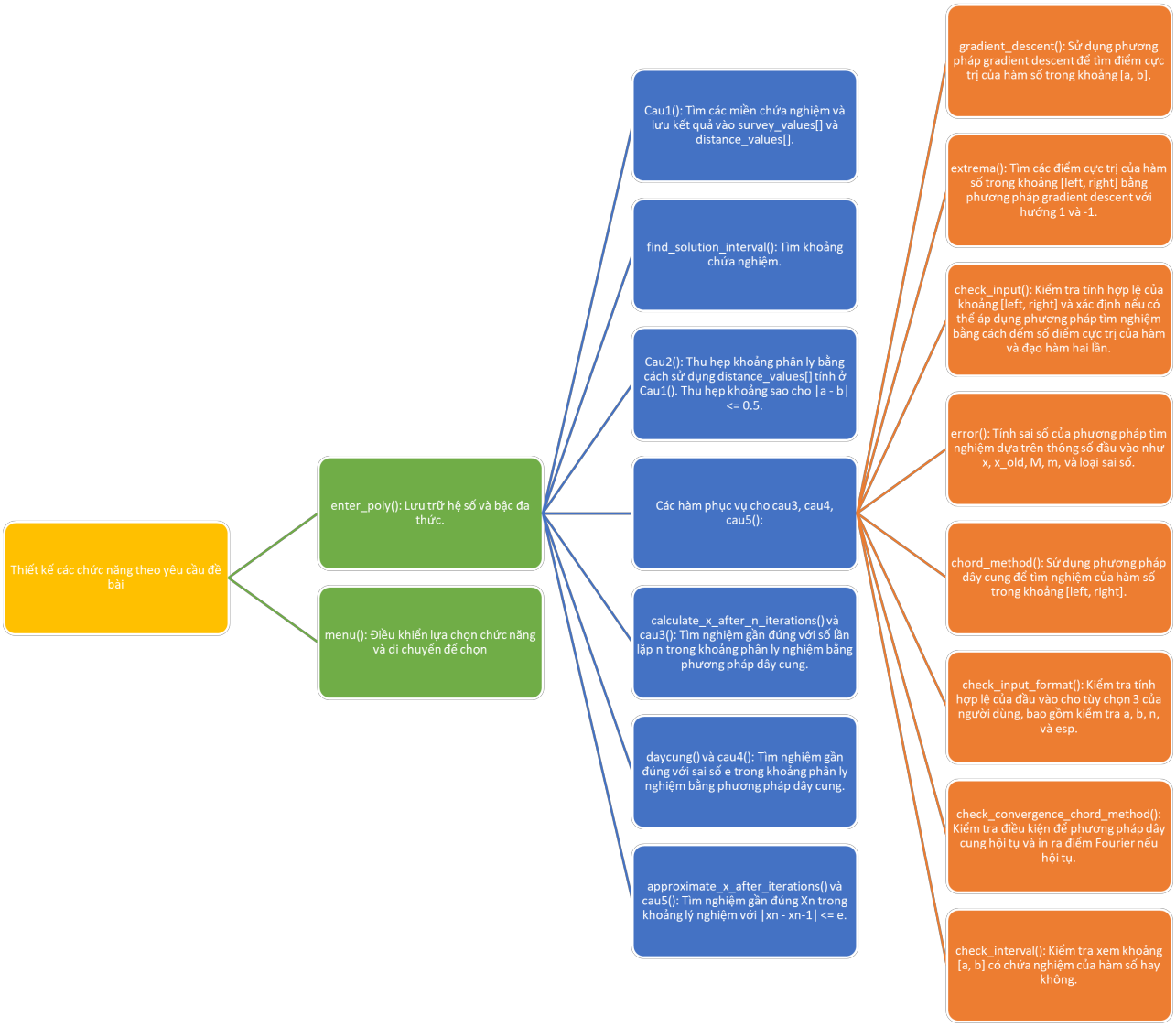
```
    TextColor(15);
    for (int i = 0; i <= 14; i++)
    {
        gotoxy(0, 14 - i); printf("%c", 186);
        gotoxy(0, 14 + i); printf("%c", 186);
        gotoxy(118, 14 - i); printf("%c", 186);
        gotoxy(118, 14 + i); printf("%c", 186);
    }
    gotoxy(0, 0); printf("%c", 201);
    gotoxy(118, 0); printf("%c", 187);
    gotoxy(0, 29); printf("%c", 200);
    gotoxy(118, 29); printf("%c", 188);

    gotoxy(88, 0); TextColor(206); printf(" KET QUA ");
    gotoxy(88, 25);TextColor(236); printf("   /\\_/\\ ");
    gotoxy(88, 26);TextColor(236); printf("  ( o.o )");
    gotoxy(88, 27);TextColor(236); printf("   > ^ < ");
    gotoxy(86, 28); TextColor(206); printf("NGUYEN VAN TUAN");

}
```

Tiếp theo là tạo các lựa chọn để sử dụng chương trình:

```
    #define MAX 7
    #define MIN 1
    const char* option[] =
{
    "Tim cac mien chua nghiem cua phuong trinh",
    "Thu hep khoang phan ly ",
    "Tim nghiem gan dung voi so lan lap n",
    "Tim nghiem gan dung voi sai so e",
    "Tim nghiem gan dung Xn",
    "Help",
    "Exit"
};
```

Bây giờ là phần thiết kế để chạy những phần đã yêu cầu, điều này đòi hỏi cần phải có những kiến thức về giải tích số.

2.2 Thiết kế các chức năng theo yêu cầu đề bài



2.2.1 Hàm `enter_poly`

Đề bài yêu cầu là giải bằng phương pháp dây cung với phương trình f(x) là đa thức bậc n.Thế nên trước khi đi vào việc giải từng phần của đề bài thì phải nhập các bậc và hệ số của đa thức đó trước. Thế nên em đã khai báo hàm enter_input(). Ở đây em sẽ không viết code mà nói ý tưởng:

Bước 1: Khởi tạo biến và giao diện

- Khai báo biến `valid_input` để kiểm tra tính hợp lệ của thông tin nhập vào.
- Hiển thị giao diện cho người dùng với yêu cầu nhập bậc của đa thức.

Bước 2: Nhập bậc của đa thức

- Sử dụng `scanf` để nhập giá trị của bậc đa thức vào biến `polynomial_degree`.
- Kiểm tra tính hợp lệ của giá trị vừa nhập:

- Nếu giá trị không là số nguyên dương hoặc vượt quá giới hạn của `N`, in thông báo lỗi và yêu cầu nhập lại.

Bước 3: Nhập các hệ số của đa thức

- Hiển thị giao diện yêu cầu nhập các hệ số của đa thức theo thứ tự.
- Sử dụng `scanf` để nhập giá trị của các hệ số vào mảng `polynomial_coefficients`.
- Kiểm tra tính hợp lệ của các hệ số:
- Nếu hệ số không phải là số thực, in thông báo lỗi và yêu cầu nhập lại.

Bước 4: Yêu cầu và nhập số lượng chữ số sau dấu phẩy

- Yêu cầu người dùng nhập số lượng chữ số sau dấu phẩy mà họ muốn hiển thị.
- Sử dụng `scanf` để nhập giá trị số lượng chữ số vào biến `digit_count`.

Bước 5: Lưu thông tin vào tệp "ketqua.txt"

- Mở tệp "ketqua.txt"bằng `fopen` để ghi thông tin vào tệp.
- Nếu không mở được tệp, thông báo lỗi.
- Sử dụng `fprintf` để ghi thông tin về bậc của đa thức và các hệ số của đa thức vào tệp.
- Đóng tệp sau khi ghi xong.

Bước 6: Lặp lại nếu có lỗi nhập

- Nếu có lỗi nhập trong quá trình nhập bậc hoặc các hệ số của đa thức, thực hiện lại quá trình nhập từ đầu (bước 1 đến bước 5) cho đến khi thông tin nhập đầy đủ và hợp lệ.

Giả mã:

```
FUNCTION enter_poly()
    SET valid_input = False
    WHILE NOT valid_input DO
        PRINT "Nhap bac cua da thuc: "
        READ polynomial_degree
        IF polynomial_degree < 0 OR polynomial_degree > N THEN
            PRINT "ERROR. Hay nhap lai."
            WAIT_FOR_A_WHILE
            CONTINUE
        END IF

        FOR i = 0 TO polynomial_degree DO
            PRINT "Nhap he so a" + i + " cua da thuc: "
            READ polynomial_coefficients[i]
            IF INPUT_IS_NOT_A_NUMBER THEN
                PRINT "He so khong hop le. Vui long nhap lai!"
                WAIT_FOR_A_WHILE
                valid_input = False
                BREAK
            END IF
        END FOR

        IF valid_input THEN
```

```
            PRINT "Ban muon hien thi may so sau dau phay: "
            READ digit_count

            PRINT "Ban muon hien thi may so sau dau phay: " + digit_count
            PRINT "Bac cua da thuc: " + polynomial_degree
            PRINT "Cac he so cua da thuc:"
            FOR i = 0 TO polynomial_degree DO
                PRINT "He so a" + i + ": " + FORMAT(polynomial_coefficients[
    i], digit_count)
            END FOR
        END IF
    END WHILE
END FUNCTION
```

2.2.2 Hàm `find_solution_interval()`

Ý tưởng của hàm `find_solution_interval()`:

Bước 1: Khởi tạo biến và tạo bản sao của các hệ số đa thức.

- Khởi tạo các biến `temp[N]`, `max = 0`, `k = 0` để lưu trữ các giá trị tạm thời và giá trị lớn nhất.
- Khởi tạo biến `upper_bound = -1` và `lower_bound = -1` để lưu giá trị đầu và cuối của khoảng chứa nghiệm.
- Khởi tạo biến `invertCoefficients = false` để kiểm tra trạng thái của việc lật dấu của các hệ số đa thức.
- Tạo bản sao của các hệ số đa thức vào mảng `temp[]`.

Bước 2: Tìm khoảng chứa nghiệm.

- Sử dụng vòng lặp `do-while` để thực hiện việc tìm khoảng chứa nghiệm.
- Lật dấu của các hệ số trong mảng `temp[]` nếu cần thiết.
- Tìm chỉ số `k` đầu tiên sao cho `temp[k] < 0`, là vị trí đầu tiên mà hàm số có giá trị âm.
- Tìm giá trị tuyệt đối lớn nhất của hệ số âm trong mảng `temp[]` và lưu vào `max`.
- Dựa vào `k` và `max`, tính giá trị `lower_bound` và `upper_bound`.

Giả mã:

```
    FUNCTION find_solution_interval_R(polynomial_coefficients: array[N] of
   Double, polynomial_degree: Integer): Double
    max = AbsoluteValue(polynomial_coefficients[0])
    R = 1 + (max / AbsoluteValue(polynomial_coefficients[0]))
    RETURN R
END FUNCTION

PROCEDURE find_solution_interval()
    DECLARE temp: array[N] of Double
    max = 0, k = 0
    upper_bound = -1, lower_bound = -1
    invertCoefficients = False
```

```
12
13      FOR i = 0 TO polynomial_degree DO
14          temp[i] = polynomial_coefficients[i]
15      END FOR
16
17      REPEAT
18          max = 0
19          k = 0
20          IF invertCoefficients THEN
21              FOR i = 0 TO polynomial_degree DO
22                  temp[i] = -temp[i]
23              END FOR
24          END IF
25
26          FOR i = 1 TO polynomial_degree DO
27              IF temp[i] < 0 THEN
28                  k = i
29                  EXIT FOR
30              END IF
31          END FOR
32
33          FOR i = 1 TO polynomial_degree DO
34              IF temp[i] < 0 AND AbsoluteValue(temp[i]) > max THEN
35                  max = AbsoluteValue(temp[i])
36              END IF
37          END FOR
38
39          IF max = 0 THEN
40              IF upper_bound = -1 THEN
41                  upper_bound = 0
42              ELSE
43                  lower_bound = 0
44              END IF
45          ELSE
46              IF upper_bound = -1 THEN
47                  upper_bound = 1 + Power((max / temp[0]), 1.0 / k)
48              ELSE
49                  lower_bound = -(1 + Power((max / temp[0]), 1.0 / k))
50              END IF
51          END IF
52
53          FOR i = 1 TO polynomial_degree STEP 2 DO
54              temp[i] = -temp[i]
55          END FOR
56
57          invertCoefficients = True
58      UNTIL lower_bound > 0
59  END PROCEDURE
```

2.2.3 Hàm `cau1()`

Ý tưởng của hàm `cau1()`:

Bước 1: Khởi tạo biến và giao diện.

- Khai báo biến và biến đếm `x0, x1, sign, temp0, temp1, value1, value2, eta, k, stt`.

- Khởi tạo biến `eta = 1e-11` để sử dụng trong phép lặp.
- Khai báo mảng `survey_values[]` và `distance_values[]` để lưu các giá trị nghiệm và các khoảng chứa nghiệm.
- Xóa màn hình và hiển thị giao diện cho người dùng.

Bước 2: Tìm khoảng chứa nghiệm của phương trình đa thức.

- Gọi hàm `find_solution_interval()` để tìm khoảng chứa nghiệm của đa thức và lưu vào biến `lower_bound` và `upper_bound`.
- Kiểm tra các trường hợp đặc biệt, nếu đa thức bậc 0 hoặc có một nghiệm duy nhất thì in kết quả thích hợp.

Bước 3: Tìm nghiệm gần đúng của phương trình.

- Sử dụng phương pháp Newton-Raphson để tìm nghiệm gần đúng của phương trình trong từng khoảng chứa nghiệm.
- Ghi thông tin và kết quả vào tệp `"ketqua.txt"` và hiển thị lên màn hình.

Bước 4: Tìm khoảng thu hẹp và ghi thông tin vào tệp `"ketqua.txt"`.

- Dựa vào các nghiệm đã tìm được, tìm các khoảng thu hẹp chứa nghiệm.
- Lưu thông tin các khoảng chứa nghiệm vào mảng `distance_values[]` và ghi vào tệp `"ketqua.txt"`.

Bước 5: Kết thúc.

- In thông báo hoàn thành và đóng tệp `"ketqua.txt"`.

Giả mã:

```
FUNCTION cau1()
    DECLARE x0, x1, sign, temp0, temp1, value1, value2: Double
    DECLARE eta = 1e-11: Double
    DECLARE k = 1, stt = 1: Integer

    giao_dien()
    find_solution_interval()

    IF polynomial_degree == 0 THEN
        IF polynomial_coefficients[0] == 0 THEN
            PRINT "Phuong trinh vo so nghiem"
        ELSE
            PRINT "Phuong trinh vo nghiem"
        END IF
    ELSE IF lower_bound == upper_bound THEN
        IF f(lower_bound) == 0 THEN
            PRINT "Nghiem cua phuong trinh la: " + lower_bound
        ELSE
            PRINT "Phuong trinh vo nghiem"
        END IF
    ELSE
        x1 = lower_bound
        WHILE x1 < upper_bound DO
            x0 = x1
            temp0 = df(x0)
            sign = IF temp0 < 0 THEN -1 ELSE 1 END IF
```

```
            x1 = x0 + sign * eta * temp0
            temp1 = df(x1)

            FOR i = 0 TO max_loop DO
                IF temp0 * temp1 > 0 THEN
                    WHILE eta < 0.008 DO
                        eta *= 2
                        x1 = x0 + sign * eta * temp0

                        IF df(x1) * temp0 < 0 THEN
                            eta /= 2
                            BREAK
                        END IF
                    END WHILE
                ELSE
                    WHILE eta > 0 DO
                        eta /= 2
                        x1 = x0 + sign * eta * temp0

                        IF df(x1) * temp0 > 0 THEN
                            BREAK
                        END IF
                    END WHILE
                END IF

                x1 = x0 + sign * eta * temp0
                x0 = x1

                IF abs(df(x1)) < 1e-4 THEN
                    survey_values[k++] = x1
                    BREAK
                END IF

                eta = 1e-11
                temp0 = df(x0)
                x1 = x0 + sign * eta * temp0
                temp1 = df(x1)

                IF x1 > upper_bound THEN
                    BREAK
                END IF
            END FOR

            x1 += 0.001
        END WHILE

        survey_values[0] = lower_bound
        survey_values[k] = upper_bound
        R = find_solution_interval_R(polynomial_coefficients,
    polynomial_degree)
        PRINT "fx nam trong mat tron tam O, R: " + R
        PRINT "Mien chua nghiem la: "
        PRINT "Can duoi: " + lower_bound
        PRINT "Can tren: " + upper_bound
    END IF

    DECLARE j = 0: Integer
    FOR i = 0 TO k DO
        value1 = f(survey_values[i])
        value2 = f(survey_values[i + 1])
```

```
86          IF value1 * value2 < 0 THEN
87              distance_values[j++] = survey_values[i]
88              distance_values[j++] = survey_values[i+1]
89              PRINT "Khoang cach ly thu " + stt
90              PRINT "Can duoi: " + survey_values[i]
91              PRINT "Can tren: " + survey_values[i + 1]
92              stt++
93          END IF
94      END FOR
95
96      IF stt == 1 THEN
97          PRINT "Phuong trinh vo nghiem"
98          RETURN
99      END IF
100 END FUNCTION
```

2.2.4 Hàm `Cau2()`

Ý tưởng của hàm `cau2()`:

Bước 1: Khởi tạo biến và giao diện.

- Khai báo các biến `i, k, stt` để lưu trạng thái và đếm các vòng lặp.
- Khai báo biến `temp` để lưu trữ giá trị trung gian trong quá trình thu hẹp khoảng.
- Khai báo biến `a, b` để lưu giá trị ban đầu của khoảng chứa nghiệm.
- Xóa màn hình và hiển thị giao diện cho người dùng.

Bước 2: Nhập thông tin và lựa chọn.

- Yêu cầu người dùng nhập số nguyên `k`, đại diện cho số thứ tự của khoảng chứa nghiệm cần thu hẹp.
- Gán giá trị cho `a` và `b` dựa vào `k` và mảng `distance_values[]` chứa các khoảng chứa nghiệm đã tìm thấy trước đó.

Bước 3: Thu hẹp khoảng chứa nghiệm.

- Sử dụng phương pháp chia đôi (bisection method) để thu hẹp khoảng chứa nghiệm.
- Lặp lại việc chia đôi cho đến khi khoảng `(b - a)` còn lớn hơn 0.5.
- Trong từng vòng lặp, tính giá trị trung bình `c` của `a` và `b`, sau đó tính giá trị của hàm số tại `c` và `a`.
- Dựa vào kết quả của hàm số tại `temp` và `a`, thu hẹp khoảng chứa nghiệm `(a, b)` bằng cách gán `c` cho `b` hoặc `a` tùy thuộc vào điều kiện.
- Ghi thông tin về các khoảng chứa nghiệm sau mỗi vòng lặp vào tệp `"ketqua.txt"`.

Bước 4: Hiển thị kết quả lên màn hình.

- Hiển thị thông tin về khoảng chứa nghiệm ban đầu và sau khi thu hẹp lên màn hình.
- Kiểm tra và hiển thị thông báo nếu không có khoảng chứa nghiệm.

Bước 5: Kết thúc.

- In thông báo hoàn thành và đóng tệp "ketqua.txt".

Giả mã:

```
FUNCTION cau2()
    DECLARE i = 0, k, stt = 1: Integer
    DECLARE c = 0, a, b, f1, f2: Double

    giao_dien()
    READ k
    a = distance_values[2 * k - 2]
    b = distance_values[2 * k - 1]

    WHILE (b - a) > 0.5 DO
        c = (a + b) / 2.0
        f1 = f(a)
        f2 = f(c)
        IF f1 * f2 < 0 THEN
            b = c
        ELSE
            a = c
        END IF
        PRINT "Lan lap thu " + stt
        PRINT "Can duoi: " + a
        PRINT "Can tren: " + b
        stt++
    END WHILE

    giao_dien_kq()
    PRINT "Khoang cach ly ban dau la:"
    PRINT "Can duoi: " + distance_values[2 * k - 2]
    PRINT "Can tren: " + distance_values[2 * k - 1]
    PRINT "Khoang cach ly sau khi rut gon la:"
    PRINT "Can duoi: " + a
    PRINT "Can tren: " + b
    IF (distance_values[2 * k - 2] == 0 AND distance_values[2 * k - 1] == 0)
    OR a == 0 OR b == 0 THEN
        PRINT "Khong co khoang phan ly nay"
    END IF
END FUNCTION
```

2.2.5 Các hàm phục vụ cau3,4,5

1: **Hàm `gradient_descent`:** Sử dụng phương pháp gradient descent để tìm điểm cực trị của hàm số. Hàm này thực hiện việc cập nhật giá trị của `current_x` theo công thức của phương pháp gradient descent cho đến khi đạt đến điều kiện dừng (`fabs(dfn(current_x, degree)) < delta`) hoặc vượt quá số vòng lặp tối đa (`max_loop`) hoặc vượt quá giới hạn của khoảng $[a, b]$. Sau đó, trả về giá trị `current_x`.

Giả mã:

```
FUNCTION gradient_descent(a, b, direction, degree)
    DECLARE current_x = a: Double
    DECLARE count = 1: Integer

```

```
    WHILE NOT (fabs(dfn(current_x, degree)) < delta) DO
        current_x = current_x - direction * learning_rate * dfn(current_x
    , degree)
        count += 1

        IF count > max_loop OR current_x > b OR current_x < a THEN
            BREAK
        END IF
    END WHILE

    RETURN current_x
END FUNCTION
```

2: Hàm `extrema`: Tìm các điểm cực trị của hàm số trong khoảng $[left, right]$ bằng cách sử dụng các lần lặp của phương pháp gradient descent với hướng 1 và -1. Các điểm cực trị được lưu vào mảng `extreme_values` và trả về mảng này.
Giả mã:

```
FUNCTION extrema(left, right, degree)
    DECLARE count = 2: Integer
    DECLARE temp1, temp2: Double
    DECLARE k = 0: Integer
    extreme_values[k++] = left
    extreme_values[k++] = right

    WHILE true DO
        temp1 = gradient_descent(left, right, 1, degree)
        temp2 = gradient_descent(left, right, -1, degree)

        IF (temp1 > right AND temp2 < left) OR (temp1 < left AND temp2 >
    right) THEN
            BREAK
        END IF

        IF temp1 > left AND temp1 < right THEN
            extreme_values[k++] = temp1
            left = extreme_values[count++] + 0.05
        END IF

        IF temp2 > left AND temp2 < right THEN
            extreme_values[k++] = temp2
            left = extreme_values[count++] + 0.05
        END IF
    END WHILE

    IF count > 2 AND degree == 1 THEN
    END IF

    RETURN extreme_values
END FUNCTION
```

3: Hàm `check_input`: Kiểm tra tính hợp lệ của khoảng $[left, right]$ và kiểm tra điều kiện để có thể áp dụng phương pháp tìm nghiệm. Hàm này gọi hai lần hàm `extrema` để tìm các điểm cực trị của hàm số và kiểm tra số lượng điểm cực trị của $f(x)$ và $f''(x)$. Nếu số lượng điểm cực trị của cả hai hàm bằng 2 thì trả về `true`, ngược lại trả về `false`.
Giả mã:

```
FUNCTION check_input(left, right)
    IF f(left) * f(right) > 0 THEN
```

```
        RETURN false
    END IF

    IF left > right THEN
        Swap left and right
    END IF

    DECLARE ext1 = extrema(left, right, 1)
    DECLARE ext2 = extrema(left, right, 2)
    DECLARE len1 = sizeof(ext1) / sizeof(ext1[0])
    DECLARE len2 = sizeof(ext2) / sizeof(ext2[0])

    IF len1 == 2 AND len2 == 2 THEN
        RETURN true
    ELSE IF len1 > 2 THEN
        PRINT "f'(x) doi dau tai it nhat 1 diem "
        RETURN false
    ELSE IF len2 > 2 THEN
        PRINT "f''(x) doi dau tai it nhat 1 diem "
        RETURN false
    END IF
END FUNCTION


```

4: Hàm `error`: Tính sai số của phương pháp tìm nghiệm dựa trên các thông số đầu vào như x, x_old, M, m, và `error_type`. Hàm này được sử dụng để tính sai số tương đối và sai số tuyệt đối.

Giả mã:

```
FUNCTION error(x, x_old, M, m, error_type)
    IF error_type == 1 THEN
        RETURN AbsoluteValue(f(x)) / m
    ELSE IF error_type == 2 THEN
        RETURN (M - m) * AbsoluteValue(x - x_old) / m
    END IF
END FUNCTION

```

5: Hàm `chord_method`: Sử dụng phương pháp dây cung (chord method) để tìm nghiệm của hàm số trong khoảng $[left, right]$. Hàm này thực hiện việc cập nhật giá trị của x theo công thức của phương pháp dây cung cho đến khi đạt đến điều kiện dừng (`fabs(f(x2)) < delta`) hoặc vượt quá số vòng lặp tối đa (`max_loop`) hoặc vượt quá giới hạn của khoảng $[left, right]$. Sau đó, trả về giá trị `x2`.

Giả mã:

```
FUNCTION chord_method(left, right, deg)
    DECLARE x1 = left, x2 = right, x: Double
    DECLARE count = 1: Integer

    WHILE NOT (fabs(f(x2)) < delta) DO
        x = x2 - (f(x2) * (x2 - x1)) / (f(x2) - f(x1))
        count += 1

        IF count > max_loop THEN
            BREAK
        ELSE IF x > right THEN
            BREAK
        ELSE IF x < left THEN
```

```
            BREAK
        END IF

        x1 = x2
        x2 = x
    END WHILE

    RETURN x2
END FUNCTION
```

6: **Hàm `check_input_format`:** Kiểm tra tính hợp lệ của đầu vào cho tùy chọn 3 của người dùng. Hàm này kiểm tra xem các giá trị a, b, n, và `esp` có hợp lệ hay không dựa trên các ràng buộc nhất định.

Giả mã:

```
FUNCTION check_input_format(a, b, n, esp)
    DECLARE flag = true: Boolean
    DECLARE aStr[100], bStr[100]: String
    Format a to aStr with precision 15
    Format b to bStr with precision 15

    IF aStr contains ',' OR bStr contains ',' THEN
        PRINT "Ban da nhap sai. Vui long nhap lai"
        flag = false
        Sleep(2000)
        Clear the screen and show the main interface
    END IF

    IF n <= 0 THEN
        PRINT "Ban da nhap sai. Vui long nhap lai"
        flag = false
        Sleep(2000)
        Clear the screen and show the main interface
    END IF

    IF esp <= 0 THEN
        PRINT "Ban da nhap sai. Vui long nhap lai"
        flag = false
        Sleep(2000)
        Clear the screen and show the main interface
    END IF

    IF NOT flag THEN
        Get a key press
        RETURN
    END IF
END FUNCTION
```

7: **Hàm `check_convergence_chord_method`:** Kiểm tra điều kiện để phương pháp dây cung hội tụ. Nếu phương pháp hội tụ, in ra thông báo về điểm Fourier (d).

Giả mã:

```
FUNCTION check_convergence_chord_method(a, b)
    IF (f(b) * df2(b) < 0 AND f(a) * df2(a) > 0) OR (f(a) * df2(a) < 0
   AND f(b) * df2(b) > 0) THEN
        DECLARE d = IF f(a) * df2(a) > 0 THEN a ELSE b END IF
        PRINT "Diem Fourier: " + d
    ELSE
```

```
        PRINT "Khong thoa man phuong phap day cung."
        PRINT "Ban da nhap sai. Vui long nhap lai"
        Sleep(2000)
        Clear the screen and show the main interface
    END IF
END FUNCTION

```

8: Hàm `check_interval`: Kiểm tra xem khoảng $[a, b]$ có chứa nghiệm của hàm số không. Nếu có, trả về `true`, ngược lại trả về `false`.
Giả mã:

```
FUNCTION check_interval(a, b)
    IF f(a) * f(b) >= 0 OR df(a) * df(b) < 0 THEN
        RETURN false
    END IF

    RETURN true
END FUNCTION

```

2.2.6 Hàm để tính cho cau3

Ý tưởng của hàm calculate_x_after_n_iterations:

Bước 1: Tính giá trị các điểm cực trị của hàm số trong khoảng $[a, b]$ bằng cách gọi hàm `extrema(a, b, 3)` và lưu vào mảng `ext`.

Bước 2: Tính giá trị của m là giá trị nhỏ nhất của hàm số đạo hàm $|f'(x)|$ trong khoảng $[a, b]$.

Bước 3: Tính giá trị của M là giá trị lớn nhất trong mảng `valExt`, chứa giá trị $|f''(x)|$ tại các điểm cực trị của hàm số.

Bước 4: Kiểm tra điều kiện $f(a) \cdot f''(a) > 0$. Nếu điều kiện thỏa mãn, đổi chỗ a và b để thỏa mãn điều kiện trên.

Bước 5: Khởi tạo các biến `x_old`, `x`, `Delta`, và `count` để sử dụng trong vòng lặp.

Bước 6: Mở tệp "ketqua.txt"bằng `fopen` để ghi kết quả vào tệp.

Bước 7: Ghi thông tin về bài toán vào tệp "ketqua.txt".

Bước 8: Trong vòng lặp, thực hiện tìm nghiệm gần đúng bằng phương pháp dây cung với hàm `chord_method` cho đến khi đạt đến sai số ε hoặc vượt quá số lần lặp tối đa.

Bước 9: Ghi kết quả nghiệm vào tệp "ketqua.txt".

Bước 10: Đóng tệp sau khi ghi xong.

Giả mã:

```
FUNCTION calculate_x_after_n_iterations(a, b, n, num, eps)
    DECLARE valExt[N]: Double
    DECLARE Delta: Double
    DECLARE k = 0: Integer
    DECLARE temp, count: Integer
    DECLARE ext = extrema(a, b, 3)
    DECLARE length = sizeof(ext) / sizeof(ext[0]): Integer

    FOR i = 0 TO length - 1 DO
        valExt[k++] = fabs(df2(ext[i]))
    END FOR

    DECLARE m = min(fabs(df(a)), fabs(df(b))): Double
    DECLARE M = max(valExt): Double

    IF f(a) * df2(a) > 0 THEN
        temp = a
        a = b
        b = temp
    END IF

    DECLARE x_old = b: Double
    DECLARE x = chord_method(a, b, 1): Double
    Delta = error(x, x_old, M, m, num)
    count = 1

    WHILE NOT (Delta < eps) AND (count < max_loop) DO
        x_old = x
        x = chord_method(a, b, 1)
        Delta = error(x, x_old, M, m, num)
        count++
    END WHILE

    FOR cnt = 1 TO n DO
        x = a - f(a) * (a - b) / (f(a) - f(b))
        a = x
        DECLARE fa = f(a)
        DECLARE fb = f(b)
    END FOR
END FUNCTION
```

Ý tưởng của hàm cau3:

Bước 1: Nhập các thông số đầu vào như khoảng cách ly nghiệm $[a, b]$, số lần lặp n, và sai số ε.

Bước 2: Kiểm tra tính hợp lệ của các thông số đầu vào bằng hàm `check_input_format`.

Bước 3: Mở tệp "ketqua.txt"bằng `fopen` để ghi kết quả vào tệp.

Bước 4: Kiểm tra tính hợp lệ của khoảng $[a, b]$ bằng hàm `check_input` và hàm `check_interval`. Nếu khoảng không hợp lệ, in thông báo lỗi và thoát.

Bước 5: Tính giá trị điểm Fourier d dựa trên điều kiện $f(a) \cdot f''(a) > 0$.

Bước 6: Kiểm tra tính hội tụ của phương pháp dây cung bằng hàm `check_convergence_chord_method`. Nếu không hội tụ, in thông báo lỗi và thoát.

Bước 7: Hiển thị giao diện yêu cầu người dùng nhập lựa chọn.

Bước 8: Nếu người dùng chọn `choose == 1`, thực hiện hàm `calculate_x_after_n_iterations` để tìm nghiệm gần đúng trong n lần lặp với số điểm lấy gần đúng là 3.

Bước 9: Ghi kết quả nghiệm vào tệp "ketqua.txt".

Bước 10: Đóng tệp sau khi ghi xong.

Giả mã:

```
FUNCTION cau3()
    DECLARE a, b, eps: Double
    DECLARE choose: Integer
    DECLARE n: Integer
    DECLARE check: Boolean
    system("cls")
    giao_dien()
    PRINT "Nhap khoang cach ly nghiem: "
    INPUT a
    PRINT "Nhap a: "
    INPUT b
    PRINT "Nhap b: "
    INPUT n
    check_input_format(a, b, n, eps)

    check = check_input(a, b)
    isIntervalValid = check_interval(a, b)

    IF NOT isIntervalValid THEN
        PRINT "Khoang phan ly a va b khong dung."
        PRINT "Hay nhap lai a va b."
        RETURN
    END IF

    d = (f(a) * df2(a) > 0) ? a : b
    check_convergence_chord_method(a, b)

    IF NOT check THEN
        PRINT "Kiem tra lai input"
        RETURN
    END IF

    PRINT "Nhap lua chon: "
    INPUT choose

    IF choose == 1 THEN
        calculate_x_after_n_iterations(a, b, n, 3, eps)
    END IF
END FUNCTION
```

2.2.7 Các hàm để tính cho cau4

Ý tưởng của hàm daycung:

Bước 1: Tính giá trị các điểm cực trị của hàm số trong khoảng $[left, right]$ bằng cách gọi hàm `extrema(left, right, 3)` và lưu vào mảng `ext`.

Bước 2: Tính giá trị của m là giá trị nhỏ nhất của hàm số đạo hàm $|f'(x)|$ trong khoảng $[left, right]$.

Bước 3: Tính giá trị của M là giá trị lớn nhất trong mảng `valExt`, chứa giá trị $|f''(x)|$ tại các điểm cực trị của hàm số.

Bước 4: Kiểm tra điều kiện $f(left) \cdot f''(left) > 0$. Nếu điều kiện thỏa mãn, đổi chỗ $left$ và $right$ để thỏa mãn điều kiện trên.

Bước 5: Khởi tạo các biến `x_old`, `x`, `Delta`, và `count` để sử dụng trong vòng lặp.

Bước 6: Mở tệp "ketqua.txt"bằng `fopen` để ghi kết quả vào tệp.

Bước 7: Ghi thông tin về bài toán vào tệp "ketqua.txt".

Bước 8: Trong vòng lặp, thực hiện tìm nghiệm gần đúng bằng phương pháp dây cung với hàm `chord_method` cho đến khi đạt đến sai số ε hoặc vượt quá số lần lặp tối đa.

Bước 9: Ghi kết quả nghiệm và số lần lặp vào tệp "ketqua.txt".

Bước 10: Đóng tệp sau khi ghi xong.

Bước 11: Khởi tạo mảng `result` và gán giá trị của `x` và `count` vào các phần tử tương ứng.

Bước 12: Trả về con trỏ `result`.

Giả mã:

```
FUNCTION daycung(left, right, num, eps)
    DECLARE valExt[N]: Array of Double
    DECLARE k, temp, count: Integer
    DECLARE ext: Pointer to Double
    ext = extrema(left, right, 3)
    DECLARE length: Integer
    length = sizeof(ext) / sizeof(ext[0])

    FOR i = 0 TO length - 1
        valExt[k++] = fabs(df2(ext[i]))

    m = min(fabs(df(left)), fabs(df(right)))
    M = max(valExt)

    IF f(left) * df2(left) > 0 THEN
        temp = left
        left = right
        right = temp
    END IF

    DECLARE x_old, x: Double
    x_old = right
    x = chord_method(left, right, 1)
    Delta = error(x, x_old, M, m, num)
    count = 1

    static result[2]: Array of Double
    result[0] = x
    result[1] = count

    RETURN result
END FUNCTION
```

Ý tưởng của hàm cau4:

Bước 1: Nhập các thông số đầu vào như khoảng cách ly nghiệm $[a, b]$, sai số ε, và lựa chọn phương pháp tính nghiệm gần đúng.

Bước 2: Kiểm tra tính hợp lệ của các thông số đầu vào bằng hàm `check_input` và hàm `check_interval`. Nếu không hợp lệ, in thông báo lỗi và thoát.

Bước 3: Tính giá trị n là số lần phải lặp của phương pháp dây cung dựa trên công thức $n = \lceil \log_2 \left(\frac{b-a}{\varepsilon}\right) \rceil$.

Bước 4: Kiểm tra tính hợp lệ của khoảng $[a, b]$ bằng hàm `check_input_format`. Nếu không hợp lệ, in thông báo lỗi và thoát.

Bước 5: Tính giá trị điểm Fourier d dựa trên điều kiện $f(a) \cdot f''(a) > 0$.

Bước 6: Kiểm tra tính hội tụ của phương pháp dây cung bằng hàm `check_convergence_chord_method`. Nếu không hội tụ, in thông báo lỗi và thoát.

Bước 7: Hiển thị giao diện yêu cầu người dùng nhập lựa chọn.

Bước 8: Nếu người dùng chọn `choose == 1`, thực hiện hàm `daycung(a, b, 1, eps)` để tìm nghiệm gần đúng bằng phương pháp dây cung với công thức $|f(x)|/m$.

Bước 9: Nếu người dùng chọn `choose == 2`, thực hiện hàm `daycung(a, b, 2, eps)` để tìm nghiệm gần đúng bằng phương pháp dây cung với công thức $(M-m) \cdot |x_n - x_{n-1}|/m$.

Bước 10: Ghi kết quả nghiệm và số lần lặp vào tệp "ketqua.txt".

Bước 11: Đóng tệp sau khi ghi xong.

Giả mã:

```
FUNCTION cau4()
    DECLARE a, b, eps: Double
    DECLARE choose: Integer
    DECLARE check: Boolean
    system("cls")
    giao_dien()

    PRINT "Nhap khoang cach ly nghiem: "
    INPUT a
    PRINT "Nhap a: "
    INPUT b
    PRINT "Nhap b: "
    INPUT eps

    check = check_input(a, b)
    n = ceil(log2((b - a) / eps))
    check_input_format(a, b, n, eps)

    DECLARE x0: Array of Double
    d = (f(a) * df2(a) > 0) ? a : b

    bool isIntervalValid = check_interval(a, b)
    IF NOT isIntervalValid THEN
        PRINT "Khoang phan ly a va b khong dung."
```

```
25          PRINT "Hay nhap lai a va b."
26          RETURN
27      END IF
28
29      check_convergence_chord_method(a, b)
30
31      IF check == false THEN
32          PRINT "Kiem tra lai input"
33          RETURN
34      ELSE IF check == true THEN
35          PRINT "Nhap lua chon: "
36          INPUT choose
37          IF choose == 1 THEN
38              x0 = daycung(a, b, 1, eps)
39              PRINT "Nghiem cua phuong trinh: ", x0[0]
40              PRINT "Tinh n bang CT (b-a)/2^n: ", n
41              PRINT "Cong thuc |f(x)|/m"
42              PRINT "n la so lan phai lap"
43          ELSE IF choose == 2 THEN
44              x0 = daycung(a, b, 2, eps)
45              PRINT "Nghiem cua phuong trinh: ", x0[0]
46              PRINT "Tinh n bang CT (b-a)/2^n: ", n
47              PRINT "Cong thuc (M-m)*|xn-xn-1|/m"
48              PRINT "n la so lan phai lap"
49          END IF
50      END IF
51  END FUNCTION
```

2.2.8 Các hàm để tính cho cau5

Ý tưởng của hàm approximate_x_after_iterations:

Bước 1: Tính giá trị các điểm cực trị của hàm số trong khoảng $[a, b]$ bằng cách gọi hàm `extrema(a, b, 3)` và lưu vào mảng `ext`.

Bước 2: Tính giá trị của m là giá trị nhỏ nhất của hàm số đạo hàm $|f'(x)|$ trong khoảng $[a, b]$.

Bước 3: Tính giá trị của M là giá trị lớn nhất trong mảng `valExt`, chứa giá trị $|f''(x)|$ tại các điểm cực trị của hàm số.

Bước 4: Kiểm tra điều kiện $f(a) \cdot f''(a) > 0$. Nếu điều kiện thỏa mãn, đổi chỗ a và b để thỏa mãn điều kiện trên.

Bước 5: Khởi tạo các biến `x_old`, `x`, `Delta`, `count`, và `num` để sử dụng trong vòng lặp.

Bước 6: Mở tệp "ketqua.txt"bằng `fopen` để ghi kết quả vào tệp.

Bước 7: Ghi thông tin về bài toán vào tệp "ketqua.txt".

Bước 8: Trong vòng lặp, thực hiện tìm nghiệm gần đúng bằng phương pháp dây cung với hàm `chord_method` cho đến khi đạt đến sai số ε hoặc vượt quá số lần lặp tối đa.

Bước 9: Ghi kết quả nghiệm và số lần lặp vào tệp "ketqua.txt".

Bước 10: Hiển thị kết quả nghiệm và số lần lặp tương ứng trên giao diện console.

Bước 11: Nếu sai số ε đủ nhỏ hoặc số lần lặp đạt tới giới hạn, thoát vòng lặp và kết thúc hàm.

Bước 12: Cập nhật giá trị của `x_old`, `a`, `fa`, và `fb` để tiếp tục vòng lặp.

Bước 13: Đóng tệp sau khi ghi xong.

Giả mã:

```
FUNCTION approximate_x_after_iterations(a, b, n, eps)
    DECLARE valExt[N]: Array of Double
    DECLARE k, temp, count, num: Integer
    DECLARE ext: Pointer to Double
    ext = extrema(a, b, 3)
    DECLARE length: Integer
    length = sizeof(ext) / sizeof(ext[0])

    FOR i = 0 TO length - 1
        valExt[k++] = fabs(df2(ext[i]))

    m = min(fabs(df(a)), fabs(df(b)))
    M = max(valExt)

    IF f(a) * df2(a) > 0 THEN
        temp = a
        a = b
        b = temp
    END IF

    DECLARE x_old, x: Double
    x_old = b
    x = chord_method(a, b, 1)
    Delta = error(x, x_old, M, m, num)
    count = 1

    FOR cnt = 1 TO n
        x = a - f(a) * (a - b) / (f(a) - f(b))

        IF fabs(x - x_old) <= eps THEN
            PRINT "Dung sau ", cnt, " lan lap vi |xn - xn-1| <= ", eps
            EXIT FOR
        END IF

        x_old = x
        a = x
        double fa = f(a)
        double fb = f(b)
    END FOR

    RETURN x
END FUNCTION
```

Ý tưởng của hàm cau5:

Bước 1: Nhập các thông số đầu vào như khoảng cách ly nghiệm $[a, b]$, sai số ε, và lựa chọn phương pháp tính nghiệm gần đúng.

Bước 2: Kiểm tra tính hợp lệ của các thông số đầu vào bằng hàm `check_input` và hàm `check_interval`. Nếu không hợp lệ, in thông báo lỗi và thoát.

Bước 3: Tính giá trị điểm Fourier d dựa trên điều kiện $f(a) \cdot f''(a) > 0$.

Bước 4: Kiểm tra tính hợp lệ của khoảng $[a, b]$ bằng hàm `check_input_format`. Nếu không hợp lệ, in thông báo lỗi và thoát.

Bước 5: Tính giá trị n là số lần phải lặp của phương pháp dây cung dựa trên công thức $n = \lceil \log_2 \left(\frac{b-a}{\varepsilon} \right) \rceil$.

Bước 6: Mở tệp "ketqua.txt"bằng `fopen` để ghi kết quả vào tệp.

Bước 7: Ghi thông tin về bài toán vào tệp "ketqua.txt".

Bước 8: Hiển thị giao diện yêu cầu người dùng nhập lựa chọn.

Bước 9: Nếu người dùng chọn `choose == 1`, thực hiện hàm `approximate_x_after_iterations(a, b, n, eps)` để tìm nghiệm gần đúng.

Bước 10: Ghi kết quả nghiệm và số lần lặp vào tệp "ketqua.txt".

Bước 11: Trong vòng lặp, thực hiện tìm nghiệm gần đúng bằng phương pháp dây cung với hàm `chord_method` cho đến khi đạt đến sai số ε hoặc vượt quá số lần lặp tối đa.

Bước 12: Ghi kết quả nghiệm và số lần lặp vào tệp "ketqua.txt".

Bước 13: Hiển thị kết quả nghiệm và số lần lặp tương ứng trên giao diện console.

Bước 14: Nếu sai số ε đủ nhỏ hoặc số lần lặp đạt tới giới hạn, thoát vòng lặp và kết thúc hàm.

Bước 15: Cập nhật giá trị của `x_old`, `a`, `fa`, và `fb` để tiếp tục vòng lặp.

Bước 16: Đóng tệp sau khi ghi xong.

Giả mã:

```
FUNCTION cau5()
    DECLARE a, b, eps: Double
    DECLARE choose: Integer
    DECLARE check: Boolean
    system("cls")
    giao_dien()

    PRINT "Nhap khoang cach ly nghiem: "
    INPUT a
    PRINT "Nhap a: "
    INPUT b
    PRINT "Nhap b: "
    INPUT eps

    check = check_input(&a, &b)
    double d = (f(a) * df2(a) > 0) ? a : b

    bool isIntervalValid = check_interval(a, b)
    IF NOT isIntervalValid THEN
        PRINT "Khoang phan ly a va b khong dung."
        PRINT "Hay nhap lai a va b."
        RETURN
    END IF

    check_convergence_chord_method(a, b)
```

```
26
27      int n = ceil(log2((b - a) / eps))
28      check_input_format(a, b, n, eps)
29
30      IF check == false THEN
31          PRINT "Kiem tra lai input"
32          RETURN
33      ELSE IF check == true THEN
34          PRINT "Nhap lua chon: "
35          INPUT choose
36          IF choose == 1 THEN
37              giao_dien_kq()
38              x = approximate_x_after_iterations(a, b, n, eps)
39              PRINT "Nghiem cua phuong trinh: ", x
40              PRINT "Tinh n bang CT (b-a)/2^n: ", n
41              PRINT "Cong thuc |f(x)|/m"
42              PRINT "n la so lan phai lap"
43          END IF
44      END IF
45  END FUNCTION
```

2.2.9 Hàm menu và int main()

Ý tưởng của hàm menu:

Bước 1: Khởi tạo biến `position = 1` để theo dõi vị trí lựa chọn của người dùng trên menu.

Bước 2: Khởi tạo biến `keyPress = 0` để theo dõi sự kiện nhấn phím của người dùng.

Bước 3: Hiển thị giao diện menu và danh sách các lựa chọn trong vòng lặp cho đến khi người dùng nhấn phím Enter (mã ASCII = 13).

Bước 4: Sử dụng vòng lặp để hiển thị danh sách các lựa chọn, đồng thời in mũi tên tại vị trí đang chọn.

Bước 5: Sử dụng hàm `_getch()` để nhận lệnh nhấn phím từ người dùng.

Bước 6: Nếu người dùng nhấn mũi tên xuống hoặc phím 's' hoặc 'S' hoặc '2', di chuyển mũi tên xuống dưới danh sách lựa chọn. Nếu đang ở vị trí cuối cùng, di chuyển về vị trí đầu tiên.

Bước 7: Nếu người dùng nhấn mũi tên lên hoặc phím 'w' hoặc 'W' hoặc '8', di chuyển mũi tên lên trên danh sách lựa chọn. Nếu đang ở vị trí đầu tiên, di chuyển về vị trí cuối cùng.

Bước 8: Sau khi người dùng nhấn phím Enter, kiểm tra giá trị của biến `position` để xử lý lựa chọn tương ứng.

Bước 9: Dựa vào giá trị của `position`, thực hiện lời gọi đến các hàm xử lý tương ứng cho từng lựa chọn.

Bước 10: Sau khi xử lý lựa chọn, xóa màn hình và gọi lại hàm `menu()` để hiển thị menu tiếp tục cho người dùng.

Bước 11: Nếu người dùng chọn lựa chọn thoát (điều kiện `position == 7`), thoát khỏi chương trình.

Ý tưởng của main();

Bước 1: Khởi tạo các biến `n`, `stt = 1`, `choose`, `a`, `b`, `eps` để sử dụng trong chương trình.

Bước 2: Hiển thị giao diện nền và nhập đa thức từ người dùng bằng hàm `enter_poly()`.

Bước 3: Gọi hàm `menu()` để hiển thị menu và xử lý các lựa chọn từ người dùng.

Bước 4: Trả về 0 để kết thúc chương trình.

3 Mã nguồn

```c
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <math.h>
#include <stdbool.h>
#include <ctype.h>
#include <windows.h>
#include <unistd.h>

#define MAX 7
#define MIN 1
#define N 100
#define max_loop 1000
#define delta 0.000001
#define learning_rate 0.01

void TextColor(int x)
{
    HANDLE h = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(h, x);
}

void gotoxy(int x, int y)
{
    COORD c = { x, y };
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), c);
}

void arrowHere(int realPosition, int arrowPosition)
{
    if (realPosition == arrowPosition)
    {
        TextColor(236); printf("%c  ", 175);
    }
    else {
        TextColor(252);  printf("    ");
    }
}
const char* option[] =
{
    "Tim cac mien chua nghiem cua phuong trinh",
    "Thu hep khoang phan ly ",
    "Tim nghiem gan dung voi so lan lap n",
    "Tim nghiem gan dung voi sai so e",
    "Tim nghiem gan dung Xn",
    "Help",
    "Exit"
};

void giao_dien_ngoai()
{
  TextColor(236);
  for(int i=0; i<=70;i++){
    for(int j=1;j<29;j++){
      gotoxy(i,j);
      printf(" ");
```

```
    }
  }
    TextColor(15);
    for (int i = 0; i <= 70; i++)
    {
        gotoxy(i, 0); printf("%c", 205);
        gotoxy(i, 29); printf("%c", 205);
    }
    gotoxy(20, 0); TextColor(206); printf(" GIAI PHUONG PHAP DAY CUNG ");
    gotoxy(0, 0); printf("%c",  201);
    gotoxy(70, 0); printf("%c", 187);
    TextColor(15);
    for (int i = 0; i <= 14; i++)
    {
        gotoxy(0, 14 - i); printf("%c", 186);
        gotoxy(0, 14 + i); printf("%c", 186);
        gotoxy(70, 14 - i); printf("%c", 186);
        gotoxy(70, 14 + i); printf("%c", 186);
    }
    gotoxy(0, 0); printf("%c", 201);
    gotoxy(70, 0); printf("%c", 187);
    gotoxy(0, 29); printf("%c", 200);
    gotoxy(70, 29); printf("%c", 188);

}
void giao_dien_pp(){
TextColor(236);
  for(int i=71; i<=118;i++){
  TextColor(236); for(int j=1;j<29;j++){
      gotoxy(i,j);
      printf(" ");
    }
  }
    TextColor(15);
    for (int i = 71; i <= 118; i++)
    {
        gotoxy(i, 0); printf("%c", 205);
        gotoxy(i, 29); printf("%c", 205);
    }
    gotoxy(84, 0); TextColor(206); printf("GIOI THIEU PP DAY CUNG ");
    gotoxy(0, 0); printf("%c",  201);
    gotoxy(118, 0); printf("%c", 187);
    TextColor(15);
    for (int i = 0; i <= 14; i++)
    {
        gotoxy(0, 14 - i); printf("%c", 186);
        gotoxy(0, 14 + i); printf("%c", 186);
        gotoxy(118, 14 - i); printf("%c", 186);
        gotoxy(118, 14 + i); printf("%c", 186);
    }
    gotoxy(0, 0); printf("%c", 201);
    gotoxy(118, 0); printf("%c", 187);
    gotoxy(0, 29); printf("%c", 200);
    gotoxy(118, 29); printf("%c", 188);
    gotoxy(72, 3);TextColor(236);; printf("f(x) la da thuc bac n:");
    gotoxy(72, 4);TextColor(236); printf("f(x)= a0*x^n + a1*x^(n-1)+...+ an
   = 0,a0!= 0");
    gotoxy(72, 7);TextColor(236);printf("Uu diem cua pp:");
    gotoxy(73, 8);TextColor(236);printf("De hieu, de trien khai");
    gotoxy(73, 9);TextColor(236);printf("Co TH hoi tu nhanh");
```

```
116     gotoxy(73, 10);TextColor(236);printf("Ap dung rong rai");
117     gotoxy(72, 13);TextColor(236);printf("Nhuoc diem cua pp");
118     gotoxy(73, 14);TextColor(236);printf("Phu thuoc diem ban dau");
119     gotoxy(73, 15);TextColor(236);printf("Kho doan dinh so lan lap");
120     gotoxy(73, 16);TextColor(236);printf("Can tinh dao ham");
121     gotoxy(73, 17);TextColor(236);printf("Co the bi sai so");
122     gotoxy(73, 18);TextColor(236);printf("Co TH hoi tu cham");
123      gotoxy(88, 25);TextColor(236); printf("   /\\_/\\ ");
124     gotoxy(88, 26);TextColor(236); printf("  ( o.o )");
125     gotoxy(88, 27);TextColor(236); printf("   > ^ < ");
126     gotoxy(86, 28); TextColor(206); printf("NGUYEN VAN TUAN");
127 }
128
129 void giao_dien_ben_trong()
130 {
131   TextColor(252);
132   for(int i=8; i<57;i++){
133     for(int j=7;j<=17;j++){
134       gotoxy(i,j);
135       printf(" ");
136     }
137   }
138   gotoxy(28, 6); TextColor(244); printf(" MENU ");
139 }
140
141 void giao_dien_kq()
142 {
143   TextColor(236);
144   for(int i=71; i<=118;i++){
145   TextColor(236); for(int j=1;j<29;j++){
146       gotoxy(i,j);
147       printf(" ");
148     }
149   }
150     TextColor(15);
151     for (int i = 71; i <= 118; i++)
152     {
153         gotoxy(i, 0); printf("%c", 205);
154         gotoxy(i, 29); printf("%c", 205);
155     }
156
157     gotoxy(0, 0); printf("%c",  201);
158     gotoxy(118, 0); printf("%c", 187);
159     TextColor(15);
160     for (int i = 0; i <= 14; i++)
161     {
162         gotoxy(0, 14 - i); printf("%c", 186);
163         gotoxy(0, 14 + i); printf("%c", 186);
164         gotoxy(118, 14 - i); printf("%c", 186);
165         gotoxy(118, 14 + i); printf("%c", 186);
166     }
167     gotoxy(0, 0); printf("%c", 201);
168     gotoxy(118, 0); printf("%c", 187);
169     gotoxy(0, 29); printf("%c", 200);
170     gotoxy(118, 29); printf("%c", 188);
171
172     gotoxy(88, 0); TextColor(206); printf(" KET QUA ");
173     gotoxy(88, 25);TextColor(236); printf("   /\\_/\\ ");
174     gotoxy(88, 26);TextColor(236); printf("  ( o.o )");
175     gotoxy(88, 27);TextColor(236); printf("   > ^ < ");
```

```
    gotoxy(86, 28); TextColor(206); printf("NGUYEN VAN TUAN");

}

void giao_dienhuongdan(){
  TextColor(236);
  for(int i=71; i<=118;i++){
  TextColor(236); for(int j=1;j<29;j++){
      gotoxy(i,j);
      printf(" ");
    }
  }
    TextColor(15);
    for (int i = 71; i <= 118; i++)
    {
        gotoxy(i, 0); printf("%c", 205);
        gotoxy(i, 29); printf("%c", 205);
    }

    gotoxy(0, 0); printf("%c",  201);
    gotoxy(118, 0); printf("%c", 187);
    TextColor(15);
    for (int i = 0; i <= 14; i++)
    {
        gotoxy(0, 14 - i); printf("%c", 186);
        gotoxy(0, 14 + i); printf("%c", 186);
        gotoxy(118, 14 - i); printf("%c", 186);
        gotoxy(118, 14 + i); printf("%c", 186);
    }
    gotoxy(0, 0); printf("%c", 201);
    gotoxy(118, 0); printf("%c", 187);
    gotoxy(0, 29); printf("%c", 200);
    gotoxy(118, 29); printf("%c", 188);
    gotoxy(86, 7); TextColor(236); printf("Have a good day!");
    gotoxy(73,9);TextColor(236);printf("On The Way To Success,");
    gotoxy(88,10);TextColor(236);printf("There Is No Trace Of Lazy Men");
    gotoxy(88, 25);TextColor(236); printf("   /\\_/\\ ");
    gotoxy(88, 26);TextColor(236); printf("  ( o.o )");
    gotoxy(88, 27);TextColor(236); printf("   > ^ < ");
    gotoxy(92, 0); TextColor(206); printf(" HELP ");
    gotoxy(86, 28); TextColor(206); printf("NGUYEN VAN TUAN");
}
void giao_dien()
{
    giao_dien_ngoai();
    giao_dien_kq();
    giao_dien_ben_trong();
}
void giao_dien_nen(){
  giao_dien_ngoai();
  giao_dien_pp();
  }
double min(double a, double b){
  if (a >= b) return b;
  return a;
}

double max(double valExt[100]){
  double max = -100000;
  for (int i = 0; i < 100; i++){
```

```
    if(valExt[i] > max) max = valExt[i];
  }
  return max;
}


double polynomial_coefficients[N];   // mang luu tru he so cua da thuc
int polynomial_degree;               // mang luu tru bac cua da thuc
double survey_values[N];             // mang luu tru cac gia tri khao sat
double distance_values[N];           // mang luu tru cac khoang cach ly
double lower_bound;                  // gioi han duoi cua mien chua ngiem
double upper_bound;                  // gioi han tren cua mien chua nghiem
double extreme_values[N];            // mang luu cac gia tri cuc tri
double m;
double M;
int digit_count;

  double f(double x)
{
    double temp = 0;
    for (int i = polynomial_degree; i >= 0; i--) {
        temp +=polynomial_coefficients[polynomial_degree - i] * pow(x, i);
    }
    return temp;
}
   double df(double x)
{
    double temp = 0;
    for (int i = polynomial_degree; i >= 1; i--) {
        temp += polynomial_coefficients[polynomial_degree - i] * i * pow(x,
   i - 1);
    }
    return temp;
}
   double df2(double x){
  double temp = 0;
  for (int i = polynomial_degree; i >= 2; i--){
    temp += polynomial_coefficients[polynomial_degree - i] * i * (i-1) * pow
   (x, i - 2);
  }
  return temp;

}
  double dfn(double x, int deg){
  if (deg == 1) return df(x);
  else if (deg == 2) return df2(x);
}
   // nhap bac, he so
void enter_poly()
{
    bool valid_input = false;
    do {
        TextColor(236);
        gotoxy(4, 3); printf(" Nhap bac cua da thuc: ");

        if (scanf("%d", &polynomial_degree) != 1 || polynomial_degree < 0 ||
    polynomial_degree > N) {
            gotoxy(4, 12); TextColor(244); printf(">> ERROR. Hay nhap lai. "
   );
            Sleep(2000);
```

```
292             TextColor(14);
293             system("cls");
294             giao_dien_nen();
295             while (getchar() != '\n')
296                 continue;
297             memset(polynomial_coefficients, 0, sizeof(
    polynomial_coefficients));
298             continue;
299         }
300 
301         gotoxy(4, 5); printf(" Nhap cac he so cua da thuc: ");
302         for (int i = 0; i <= polynomial_degree; i++) {
303             gotoxy(4, 6 + i);
304             printf(">> a%d = ", i);
305             if (scanf("%lf", &polynomial_coefficients[i]) != 1) {
306                 gotoxy(4, 19); printf(">> He so khong hop le. Vui long nhap
    lai!");
307                 Sleep(2000);
308                 system("cls");
309                 giao_dien_nen();
310                 while (getchar() != '\n')
311                     continue;
312                 valid_input = false;
313                 memset(polynomial_coefficients, 0, sizeof(
    polynomial_coefficients));
314                 break;
315             }
316             valid_input = true;
317         }
318 
319         if (!valid_input)
320             continue;
321 
322         gotoxy(4, 18); printf("Ban muon hien thi may so sau dau phay: ");
323         scanf("%d", &digit_count);
324 
325         FILE* fout = fopen("ketqua.txt", "a");
326         if (fout == NULL) {
327             printf("Khong mo duoc File de ghi.\n");
328             return;
329         }
330         fprintf(fout, "Ban muon hien thi may so sau dau phay: %d\n",
    digit_count);
331         fprintf(fout, "Bac cua da thuc: %d\n", polynomial_degree);
332         fprintf(fout, "Cac he so cua da thuc:\n");
333         for (int i = 0; i <= polynomial_degree; i++) {
334             fprintf(fout, "He so a%d: %.*lf\n", i, digit_count,
    polynomial_coefficients[i]);
335         }
336 
337         fclose(fout);
338 
339     } while (!valid_input);
340 }
341 
342 //option 1
343 
344 float find_solution_interval_R(double polynomial_coefficients[N],int
    polynomial_degree) {
345     float max = fabs(polynomial_coefficients[0]);
```

```
346     float R;
347 
348     for (int i = 0; i <= polynomial_degree; i++) {
349         if (fabs(polynomial_coefficients[i]) >= max) {
350             max = fabs(polynomial_coefficients[i]);
351         }
352     }
353 
354     R = 1 + (max / fabs(polynomial_coefficients[0]));
355     return R;
356 }
357 void find_solution_interval() {
358     double temp[N],max=0,k=0;
359     upper_bound = -1;
360     lower_bound = -1;
361     bool invertCoefficients = false;
362 
363     for (int i = 0; i <= polynomial_degree; i++) {
364         temp[i] = polynomial_coefficients[i];
365     }
366 
367     do {
368         max = 0;
369         k = 0;
370         if (invertCoefficients) {
371             for (int i = 0; i <= polynomial_degree; i++) {
372                 temp[i] = -temp[i];
373             }
374         }
375 
376         for (int i = 1; i <= polynomial_degree; i++) {
377             if (temp[i] < 0) {
378                 k = i;
379                 break;
380             }
381         }
382 
383         for (int i = 1; i <= polynomial_degree; i++) {
384             if (temp[i] < 0 && fabs(temp[i]) > max) {
385                 max = fabs(temp[i]);
386             }
387         }
388 
389        (max == 0) ? ((upper_bound == -1) ? upper_bound = 0 : lower_bound =
    0)
390                    : ((upper_bound == -1) ? upper_bound = 1 + pow((max /
    temp[0]), 1.0 / k)
391                                           : lower_bound = -(1 + pow((max /
    temp[0]), 1.0 / k)));
392         for (int i = 1; i <= polynomial_degree; i += 2) {
393             temp[i] = -temp[i];
394         }
395 
396         invertCoefficients = true;
397     } while (lower_bound > 0);
398 }
399 
400 void cau1()
401 {
402     double x0, x1, sign, temp0, temp1, value1, value2;
```

```
403     double eta = 1e-11;
404     int k = 1;
405     int stt = 1;
406     system("cls");
407     giao_dien();
408      find_solution_interval();
409    if (polynomial_degree == 0) {
410     (polynomial_coefficients[0] == 0) ? (gotoxy(84, 3), TextColor(236),
    printf("Phuong trinh vo so nghiem"))
411                                       : (gotoxy(84, 3), TextColor(236),
    printf("Phuong trinh vo nghiem"));
412 }
413 else if (lower_bound == upper_bound) {
414     (f(lower_bound) == 0) ? (gotoxy(84, 3), TextColor(236), printf("Nghiem
    cua phuong trinh la: %.*lf", digit_count, lower_bound))
415                           : (gotoxy(84, 3), TextColor(236), printf("Phuong
    trinh vo nghiem"));
416 }
417 else {
418     x1 = lower_bound;
419     while (x1 < upper_bound) {
420         x0 = x1;
421         temp0 = df(x0);
422         sign = (temp0 < 0) ? -1 : 1;
423         x1 = x0 + sign * eta * temp0;
424         temp1 = df(x1);
425 
426         for (int i = 0; i < max_loop; i++) {
427             if (temp0 * temp1 > 0) {
428                 while (eta < 0.008) {
429                     eta *= 2;
430                     x1 = x0 + sign * eta * temp0;
431 
432                     if (df(x1) * temp0 < 0) {
433                         eta /= 2;
434                         break;
435                     }
436                 }
437             }
438             else {
439                 while (eta > 0) {
440                     eta /= 2;
441                     x1 = x0 + sign * eta * temp0;
442 
443                     if (df(x1) * temp0 > 0) {
444                         break;
445                     }
446                 }
447             }
448 
449             x1 = x0 + sign * eta * temp0;
450             x0 = x1;
451 
452             if (abs(df(x1)) < 1e-4) {
453                 survey_values[k++] = x1;
454                 break;
455             }
456 
457             eta = 1e-11;
458             temp0 = df(x0);
```

```
            x1 = x0 + sign * eta * temp0;
            temp1 = df(x1);

            if (x1 > upper_bound) {
                break;
            }
        }

        x1 += 0.001;
    }


        survey_values[0] = lower_bound;
        survey_values[k] = upper_bound;
        gotoxy(13, 9);TextColor(252); printf("Day la nhung mien chua nghiem");
        gotoxy(13, 11); TextColor(252);printf("Hay nho so khoang muon thu hep de lam p2");
        gotoxy(13, 13); TextColor(252);printf("Nen nho cac khoang de lam cau sau thuan tien");
        gotoxy(13, 15); TextColor(252);printf("Tranh nhap dau phay ma phai la dau cham");
        float R = find_solution_interval_R(polynomial_coefficients, polynomial_degree);
        gotoxy(76,1);TextColor(236);printf("fx nam trong mat tron tam O, R: %.*f",digit_count,R);
        gotoxy(84,2);  TextColor(236);
        printf("Mien chua nghiem la: ");
        gotoxy(84,3); TextColor(236);
        printf("Can duoi : %.*lf", digit_count, lower_bound);
        gotoxy(84,4); TextColor(236);
        printf("Can tren : %.*lf", digit_count, upper_bound);
    }
    FILE* fout = fopen("ketqua.txt", "a");
     fprintf(fout, "\n_____________________");
     fprintf(fout,"\nCau 1: Tim mien chua nghiem cua phuong trinh da thuc fx");
      float R = find_solution_interval_R(polynomial_coefficients, polynomial_degree);
      fprintf(fout,"\nfx nam trong mat tron tam O, R: %.*f",digit_count,R);
          fprintf(fout,"\nMien chua nghiem la: ");
          fprintf(fout,"\nCan duoi : %.*lf", digit_count, lower_bound);
          fprintf(fout,"\nCan tren : %.*lf", digit_count, upper_bound);
int j=0;
    for (int i = 0; i < k; i++) {
        value1 = f(survey_values[i]);
        value2 = f(survey_values[i + 1]);
        if (value1 * value2 < 0) {
          distance_values[j++] = survey_values[i];
      distance_values[j++] = survey_values[i+1];
            fprintf(fout, "\nKhoang cach ly thu %d\n", stt);
            fprintf(fout, "\nCan duoi: %.*lf\nCan tren: %.*lf\n",digit_count, survey_values[i], digit_count, survey_values[i + 1]);
            gotoxy(84, 6+3*(stt-1));
            printf("%d. Khoang cach ly thu %d", stt, stt);
            gotoxy(84, 7+3*(stt-1));
            printf("- Can duoi: %.*lf", digit_count, survey_values[i]);
            gotoxy(84, 8+3*(stt-1));
            printf("- Can tren: %.*lf", digit_count, survey_values[i+1]);
          stt++;
```

```
        }
    }if (stt == 1) {
    gotoxy(84, 5); TextColor(236);
    printf("Phuong trinh vo nghiem");
    gotoxy(84,6); TextColor(236);printf("Hay nhap lai ban nhe!!");
    fprintf(fout,"\nPhuong trinh vo nghiem");
    sleep(5);
    exit(0);

    }
    fclose(fout);
}



  // Option 2
void cau2()
{
    int i = 0, k, stt = 1;
    double c = 0,a, b, f1, f2;
    system("cls");
    giao_dien();
    gotoxy(16, 9);TextColor(252);
    printf("Thu hep khoang phan ly nghiem ");
    gotoxy(16, 10);TextColor(252);printf("Hay nhap so nguyen k=1,2,3,..,n");
    gotoxy(16, 12);TextColor(252);printf("Chon khoang phan ly nghiem k= ");
    FILE* fout = fopen("ketqua.txt", "a");
    fprintf(fout, "\n_____________________");
    fprintf(fout," \nCau 2: Thu hep khoang phan ly");
    fprintf(fout, "\nKhoang cach ly ban dau:\n");
    fprintf(fout, "\nCan duoi: %.*lf\nCan tren: %.*lf\n", digit_count, a,
   digit_count, b);
    scanf("%d", &k);
    a = distance_values[2 * k - 2];
    b = distance_values[2 * k - 1];
    while ((b - a) > 0.5) {
        c = (a + b) / 2.0;
        f1 = f(a);
        f2 = f(c);
        if (f1 * f2 < 0) {
            b = c;
        }
        else {
            a = c;
        }
        fprintf(fout, "\nLan lap thu %d\n", stt);
        fprintf(fout, "\nCan duoi: %.*lf\nCan tren: %.*lf\n", digit_count, a
   , digit_count, b);
        stt++;
    }

    TextColor(236);
    giao_dien_kq();
    gotoxy(80, 3);TextColor(236);
    printf("Khoang cach ly ban dau la:");
    gotoxy(80, 4);TextColor(236);
    printf("Can duoi: %.*lf", digit_count, distance_values[2 * k - 2]);
    gotoxy(80, 5);TextColor(236);
    printf("Can tren: %.*lf", digit_count, distance_values[2 * k - 1]);
    gotoxy(80, 7);TextColor(236);
```

```
568     printf("Khoang cach ly sau khi rut gon la:");
569     gotoxy(80, 8);TextColor(236);
570     printf("Can duoi: %.*lf", digit_count, a);
571     gotoxy(80, 9);TextColor(236);
572     printf("Can tren: %.*lf", digit_count, b);
573     if ((distance_values[2 * k - 2] == 0 && distance_values[2 * k - 1] == 0)
        || a == 0 || b == 0) {
574         gotoxy(80, 12);
575         printf("Khong co khoang phan ly nay");
576         fprintf(fout,"\nKhong co khoang phan ly nay");
577     gotoxy(80, 14);printf("Hay nhap lai nha");
578     }
579 
580     fclose(fout);
581 }
582 
583   // cac ham de thuc hien cac option con lai
584 double error(double x, double x_old, double M, double m, int error_type){
585     if (error_type == 1)
586         return fabs(f(x))/ m;
587     else if (error_type== 2)
588         return (M-m) * abs(x - x_old)/m;
589 }
590     double chord_method(double left, double right, int deg) {
591     double x1 = left;
592     double x2 = right;
593     double x;
594     int count = 1;
595 
596     while (!(fabs(f(x2)) < delta)) {
597         x = x2 - (f(x2) * (x2 - x1)) / (f(x2) - f(x1));
598         count += 1;
599 
600         if (count > max_loop) {
601             break;
602         } else if (x > right) {
603             break;
604         } else if (x < left) {
605             break;
606         }
607 
608         x1 = x2;
609         x2 = x;
610     }
611 
612     return x2;
613 }
614  double gradient_descent(double a, double b, int direction, int  degree) {
615     double current_x = a;
616     int count = 1;
617 
618     while (!(fabs(dfn(current_x,  degree)) < delta)) {
619         current_x= current_x - direction *learning_rate * dfn(current_x,
    degree);
620         count += 1;
621 
622         if (count > max_loop || current_x > b || current_x < a) {
623             break;
624         }
625     }
```

```
626
627     return current_x;
628 }
629 double* extrema(double left, double right, int  degree) {
630     int count = 2;
631     double temp1, temp2;
632     int k = 0;
633     extreme_values[k++] = left;
634     extreme_values[k++] = right;
635
636     while (1) {
637         temp1 = gradient_descent(left, right, 1,  degree);
638         temp2 = gradient_descent(left, right, -1,  degree);
639
640         if ((temp1 > right && temp2 < left) || (temp1 < left && temp2 >
    right)) {
641             break;
642         }
643
644         if (temp1 > left && temp1 < right) {
645             extreme_values[k++] = temp1;
646             left = extreme_values[count++] + 0.05;
647         }
648
649         if (temp2 > left && temp2 < right) {
650             extreme_values[k++] = temp2;
651             left = extreme_values[count++] + 0.05;
652         }
653     }
654
655     if (count > 2 &&  degree== 1) {
656         // printf("Ham so ton tai cuc tri tren khoang (%.*lf , %.*lf)",
    digit_count, digit_count, extreme_values[0], extreme_values[1]);
657     }
658
659     return extreme_values;
660 }
661
662 double check_input(double *left, double *right){
663   if(f(*left) * f(*right) > 0) return false;
664   if (*left > *right) {
665     double temp = *left;
666     *left = *right;
667     *right = temp;
668   }
669   double *ext1 = extrema(*left, *right, 1);
670   double *ext2 = extrema(*left, *right, 2);
671   int len1 = sizeof(ext1) / sizeof(ext1[0]);
672   int len2 = sizeof(ext2) / sizeof(ext2[0]);
673   if (len1 == 2 && len2 == 2) return true;
674     else if (len1 > 2){
675         printf("f'(x) doi dau tai it nhat 1 diem ");
676         return false;
677   }
678   else if (len2 > 2){
679         printf("f\"(x) doi dau tai it nhat 1 diem ");
680         return false;
681   }
682 }
683
```

```
684
685 void check_input_format(double a, double b, int n, double esp) {
686     bool flag = true;
687     char aStr[100], bStr[100];
688     sprintf(aStr, "%.15g", a);
689     sprintf(bStr, "%.15g", b);
690     if (strchr(aStr, ',') != NULL || strchr(bStr, ',') != NULL) {
691         gotoxy(80,3);TextColor(236);printf("Ban da nhap sai. Vui long nhap
    lai");
692         flag = false;
693         Sleep(2000);
694         system("cls");
695             giao_dien();
696     }
697
698     if (n <= 0) {
699         gotoxy(80,3);TextColor(236);printf("Ban da nhap sai. Vui long nhap
    lai");
700         flag = false;
701         Sleep(2000);
702         system("cls");
703         giao_dien();
704     }
705
706     if (esp <= 0) {
707         gotoxy(80,3);TextColor(236);printf("Ban da nhap sai. Vui long nhap
    lai");
708         flag = false;
709         Sleep(2000);
710         system("cls");
711          giao_dien();
712     }
713     if (!flag) {
714
715         getch();
716         return;
717     }
718 }
719
720
721
722 void check_convergence_chord_method(double a, double b) {
723     if ((f(b) * df2(b) < 0 && f(a) * df2(a) > 0) || (f(a) * df2(a) < 0 && f(
    b) * df2(b) > 0)) {
724         // diem Fourier
725         double d = (f(a) * df2(a) > 0) ? a : b;
726         gotoxy(16,13);TextColor(252);printf("Diem Fourier: %lf\n", d);
727     } else {
728          gotoxy(16,13);TextColor(252);printf("Khong thoa man phuong phap day
     cung.\n");
729           gotoxy(80,3);TextColor(236);printf("Ban da nhap sai. Vui long nhap
     lai");
730          Sleep(2000);
731         system("cls");
732          giao_dien();
733     }
734 }
735
736 bool check_interval(double a, double b) {
737     if (f(a) * f(b) >= 0) {
```

```
738         return false;
739     }
740     if (df(a)*df(b)<0){
741         return false;
742     }
743     return true;
744 }
745 // option 3
746 void calculate_x_after_n_iterations(double a, double b, int n, int num,
        double eps) {
747     double valExt[N], Delta;
748     int k = 0, temp, count;
749     double *ext = extrema(a, b, 3);
750     int length = sizeof(ext) / sizeof(ext[0]);
751 
752     for (int i = 0; i < length; i++) {
753         valExt[k++] = fabs(df2(ext[i]));
754     }
755 
756     double m = min(fabs(df(a)), fabs(df(b)));
757     double M = max(valExt);
758 
759     if (f(a) * df2(a) > 0) {
760         temp = a;
761         a = b;
762         b = temp;
763     }
764 
765     double x_old = b;
766     double x = chord_method(a, b, 1);
767     Delta = error(x, x_old, M, m, num);
768     count = 1;
769 
770     FILE *fout;
771     fout = fopen("ketqua.txt", "a");
772     fprintf(fout, "\n_______________________");
773     fprintf(fout,"\nCau 3: Tim nghiem gan dung voi so lan lap n");
774 
775     while (!(Delta < eps) && (count < max_loop)) {
776         x_old = x;
777         x = chord_method(a, b, 1);
778         Delta = error(x, x_old, M, m, num);
779         count++;
780     }
781 
782     fprintf(fout, "\nNghiem cua phuong trinh la: %lf", x);
783     fclose(fout);
784 
785     gotoxy(80, 4);TextColor(236);
786     printf("Nghiem cua phuong trinh:");
787     for (int cnt = 1; cnt <= n; cnt++) {
788         double x = a - f(a) * (a - b) / (f(a) - f(b));
789         gotoxy(80, cnt + 4);TextColor(236);
790         printf("%d. x = %.*lf\n", cnt,digit_count, x);
791         a = x;
792         double fa = f(a);
793         double fb = f(b);
794     }
795 }
796 
```

```
797 void cau3() {
798     double a, b, eps;
799     int choose;
800     int n;
801     bool check;
802     FILE *fout;
803     system("cls");
804     giao_dien();
805     TextColor(236);
806     gotoxy(16, 9);TextColor(252);
807     printf("Nhap khoang cach ly nghiem: ");
808     gotoxy(16, 10);TextColor(252);
809     printf("Nhap a: ");
810     scanf("%lf", &a);
811     gotoxy(16, 11);TextColor(252);
812     printf("Nhap b: ");
813     scanf("%lf", &b);
814     gotoxy(16, 12);TextColor(252);
815     printf("Nhap n: ");
816     scanf("%d", &n);
817 
818     check_input_format(a, b, n, eps);
819 
820     fout = fopen("ketqua.txt", "a");
821     check = check_input(&a, &b);
822     bool isIntervalValid = check_interval(a, b);
823     if (!isIntervalValid) {
824         gotoxy(80, 4);TextColor(236);
825         printf("Khoang phan ly a va b khong dung.\n");
826         gotoxy(80, 6);TextColor(236);
827         printf("Hay nhap lai a va b.\n");
828         fprintf(fout, "Khoang phan ly a va b khong dung");
829         fclose(fout);
830         return;
831     }
832 
833     double d = (f(a) * df2(a) > 0) ? a : b;
834     check_convergence_chord_method(a, b);
835     if (!check) {
836         gotoxy(16, 14);TextColor(236);
837         printf("Kiem tra lai input");
838          Sleep(2000);
839         system("cls");
840          giao_dien();
841         fclose(fout);
842         return;
843     }
844 
845     gotoxy(16, 14);TextColor(252);
846     printf("Nhap lua chon: ");
847     scanf("%d", &choose);
848 
849     fprintf(fout, "\n nhap a: %.*lf",digit_count,a);
850     fprintf(fout, "\n nhap b: %.*lf",digit_count, b);
851     fprintf(fout, "\n nhap n: %d", n);
852     fprintf(fout, "\n Diem Fourier: %.*lf\n", digit_count,d);
853     fprintf(fout, "\n So lan lap: %d", n);
854 
855     if (choose == 1) {
856         giao_dien_kq();
```

```
857         gotoxy(80, 3);TextColor(236);
858         printf("So lan lap: %d", n);
859         calculate_x_after_n_iterations(a, b, n, 3, eps);
860         fprintf(fout, "\nNghiem cua phuong trinh:\n");
861         for (int cnt = 1; cnt <= n; cnt++) {
862             double x = a - f(a) * (a - b) / (f(a) - f(b));
863             fprintf(fout, "%d. x = %.*lf\n", cnt,digit_count, x);
864             a = x;
865             double fa = f(a);
866             double fb = f(b);
867         }
868         fclose(fout);
869     }
870 }
871
872
873
874
875      // option 4
876
877 double* daycung(double left, double right, int num, double eps) {
878     double valExt[N], Delta;
879     int k = 0, temp, count;
880     double *ext = extrema(left, right, 3);
881     int length = sizeof(ext) / sizeof(ext[0]);
882
883     for (int i = 0; i < length; i++) {
884         valExt[k++] = fabs(df2(extreme_values[i]));
885     }
886
887     m = min(fabs(df(left)), fabs(df(right)));
888     M = max(valExt);
889
890     if (f(left) * df2(left) > 0) {
891         temp = left;
892         left = right;
893         right = temp;
894     }
895
896     double x_old = right;
897     double x = chord_method(left, right, 1);
898     Delta = error(x, x_old, M, m, num);
899     count = 1;
900
901     FILE *fout;
902     fout = fopen("ketqua.txt", "a");
903     fprintf(fout, "\n_____________________");
904     fprintf(fout,"\nCau 4: Tim nghiem gan dung voi sai so e");
905     while (!(Delta < eps) && (count < max_loop)) {
906         x_old = x;
907         x = chord_method(left, right, 1);
908         Delta = error(x, x_old, M, m, num);
909         count++;
910     }
911
912     fprintf(fout, "\nNghiem cua phuong trinh la: %.*lf",digit_count , x);
913     fprintf(fout, "\nSo lan lap: %d", count);
914     fclose(fout);
915
916     static double result[2];
```

```
    result[0] = x;
    result[1] = count;

    return result;
}

void cau4() {
    double a, b, eps;
    int choose;
    bool check;
    FILE *fout;
    system("cls");
    giao_dien();
    TextColor(236);
    gotoxy(16, 9);TextColor(252); printf("Nhap khoang cach ly nghiem: ");
    gotoxy(16, 10); TextColor(252);printf("Nhap a: "); scanf("%lf", &a);
    gotoxy(16, 11); TextColor(252);printf("Nhap b: "); scanf("%lf", &b);
    gotoxy(16, 12);TextColor(252); printf("Nhap sai so: "); scanf("%lf", &
   eps);
    check = check_input(&a, &b);
    int n = ceil(log2((b - a) / eps));
     check_input_format(a, b, n, eps);
    double *x0;
    double d = (f(a) * df2(a) > 0) ? a : b;
    fout = fopen("ketqua.txt", "a");
    bool isIntervalValid = check_interval(a, b);
    if (!isIntervalValid) {
       gotoxy(80,4);TextColor(236); printf("Khoang phan ly a va b khong dung
   .\n");
       gotoxy(80,6);TextColor(236); printf("Hay nhap lai a va b.\n");
       fprintf(fout,"khoang phan ly a va b khong dung");
        Sleep(2000);
        system("cls");
         giao_dien();
        return;
    }
    check_convergence_chord_method(a,b);

    if (check == false) {
        gotoxy(16, 14); TextColor(252);printf("Kiem tra lai input"); Sleep
   (2000);
        system("cls");
         giao_dien();
    } else if (check == true) {
        gotoxy(16, 14); TextColor(252);printf("Nhap lua chon: "); scanf("%d"
   , &choose);

        if (choose == 1) {
           fprintf(fout,"\n nhap a: %.*lf", digit_count,a);
             fprintf(fout,"\n nhap b: %.*lf", digit_count,b);
             fprintf(fout,"\n nhap sai so: %.*lf", digit_count,eps);
             fprintf(fout,"\n Diem Fourier: %*lf\n", digit_count,d);
             fprintf(fout,"\n nhap lua chon: %d",choose);
            x0 = daycung(a, b, 1, eps);
            giao_dien_kq();
            gotoxy(80, 4);TextColor(236); printf("Nghiem cua phuong trinh:
   %.*lf", digit_count, x0[0]);
            gotoxy(80,5);TextColor(236);printf("Tinh n bang CT (b-a)/2^n: %d
   ",n);
            gotoxy(80,3);TextColor(236);printf("Cong thuc |f(x)|/m");
```

```
971             gotoxy(80,6);TextColor(236);printf("n la so lan phai lap");
972             fprintf(fout,"\nCong thuc |f(x)|/m");
973             fprintf(fout, "\nNghiem cua phuong trinh la: %.*lf", digit_count
    , x0[0]);
974             fprintf(fout,"\n Tinh n bang CT (b-a)/2^n: %d",n);
975             fprintf(fout,"\n n la so lan phai lap");
976             fclose(fout);
977         } else if (choose == 2) {
978           fprintf(fout,"\n nhap a: %.*lf", digit_count,a);
979              fprintf(fout,"\n nhap b: %.*lf", digit_count,b);
980              fprintf(fout,"\n nhap sai so: %.*lf", digit_count,eps);
981              fprintf(fout,"\n Diem Fourier: %*lf\n", digit_count,d);
982              fprintf(fout,"\n nhap lua chon: %d",choose);
983           giao_dien_kq();
984             x0 = daycung(a, b, 2, eps);
985             fprintf(fout,"\n nhap lua chon: %d",choose);
986             gotoxy(80, 4);TextColor(236);printf("Nghiem cua phuong trinh:
    %.*lf", digit_count, x0[0]);
987             fout = fopen("ketqua.txt", "a");
988             fprintf(fout,"\nCong thuc (M-m)*|xn-xn-1|/m");
989             fprintf(fout, "\nNghiem cua phuong trinh la: %f", x0[0]);
990             fprintf(fout,"\n Tinh n bang CT (b-a)/2^n: %d",n);
991             fprintf(fout,"\n n la so lan phai lap");
992             gotoxy(80,5);TextColor(236);printf("Tinh n bang CT (b-a)/2^n: %d
    ",n);
993             gotoxy(80,6);TextColor(236);printf("n la so lan phai lap");
994             gotoxy(80,3);TextColor(236);printf("Cong thuc (M-m)*|xn-xn-1|/m"
    );
995             fclose(fout);
996         }
997     }
998 }
999
1000
1001     // option 5
1002  void approximate_x_after_iterations(double a, double b, int n, double eps)
    {
1003     double valExt[N], Delta;
1004     int k = 0, temp, count, num;
1005     double *ext = extrema(a, b, 3);
1006     int length = sizeof(ext) / sizeof(ext[0]);
1007
1008     for (int i = 0; i < length; i++) {
1009         valExt[k++] = fabs(df2(ext[i]));
1010     }
1011
1012     double m = min(fabs(df(a)), fabs(df(b)));
1013     double M = max(valExt);
1014
1015     if (f(a) * df2(a) > 0) {
1016         temp = a;
1017         a = b;
1018         b = temp;
1019     }
1020
1021     double x_old = b;
1022     double x = chord_method(a, b, 1);
1023     Delta = error(x, x_old, M, m, num);
1024     count = 1;
1025
```

```
    FILE *fout;
    fout = fopen("ketqua.txt", "a");
    fprintf(fout, "\n_________________________");
    fprintf(fout,"\nCau 5: Tm nghiem gan dung cua Xn");

    while (!(Delta < eps) && (count < max_loop)) {
        x_old = x;
        x = chord_method(a, b, 1);
        Delta = error(x, x_old, M, m, num);
        count++;
    }

    fprintf(fout, "\nNghiem cua phuong trinh la: %.*lf", digit_count, x);
    fclose(fout);

    gotoxy(80,4);TextColor(236);printf("Nghiem cua phuong trinh:");
    for (int cnt = 1; cnt <= n; cnt++) {
        x = a - f(a) * (a - b) / (f(a) - f(b));
        gotoxy(80,4+cnt);TextColor(236);printf("%d. x = %.*lf\n", cnt,
   digit_count, x);

        if (fabs(x - x_old) <= eps) {

            gotoxy(80,5+cnt);TextColor(236);printf("Dung sau %d lan lap  ",
    cnt);
            gotoxy(80,6+cnt);TextColor(236);printf("vi |xn - xn-1| <= %.9lf
   \n", eps);
          break;
        }

        x_old = x;
        a = x;
        double fa = f(a);
        double fb = f(b);
    }
}

    void cau5() {
    double a, b, eps;
    int choose;
    double x_old;
    bool check;
    FILE *fout;
    system("cls");
    giao_dien();
    gotoxy(12, 10);TextColor(252); printf("Nhap khoang cach ly nghiem: Nhap
   a: "); scanf("%lf", &a);
    gotoxy(16, 11); TextColor(252);printf("Nhap b: "); scanf("%lf", &b);
    gotoxy(16, 12);TextColor(252); printf("Nhap epsilon: "); scanf("%lf", &
   eps);
    gotoxy(16,10); TextColor(252);("Nhap khoang cach ly nghiem: ");
    check = check_input(&a, &b);
    double d = (f(a) * df2(a) > 0) ? a : b;
    fout = fopen("ketqua.txt", "a");
     bool isIntervalValid = check_interval(a, b);
    if (!isIntervalValid) {
       gotoxy(80,4);TextColor(236); printf("Khoang phan ly a va b khong dung
   .\n");
       gotoxy(80,6);TextColor(236); printf("Hay nhap lai a va b.\n");
       fprintf(fout,"khoang phan ly a va b khong dung");
```

```
1080         Sleep(2000);
1081         system("cls");
1082          giao_dien();
1083         return;
1084     }
1085     check_convergence_chord_method(a,b);
1086      int n = ceil(log2((b - a) / eps));
1087      check_input_format(a, b, n, eps);
1088      fprintf(fout,"\n nhap a:%.*lf", digit_count,a);
1089     fprintf(fout,"\n nhap b: %.*lf", digit_count,b);
1090     fprintf(fout,"\n nhap epsilon: %.*lf", digit_count,eps);
1091     fprintf(fout,"\n Diem Fourier: %.*lf\n", digit_count ,d);
1092     if (check == false) {
1093         gotoxy(16, 14);TextColor(252); printf("Kiem tra lai input"); Sleep
    (2000);
1094         system("cls");
1095          giao_dien();
1096     } else if (check == true) {
1097         gotoxy(16, 14); TextColor(252);printf("Nhap lua chon: "); scanf("%d"
    , &choose);
1098     }
1099     if (choose == 1) {
1100         giao_dien_kq();
1101         approximate_x_after_iterations(a, b, n, eps);
1102         fprintf(fout, "\nNghiem cua phuong trinh:\n");
1103         for (int cnt = 1; cnt <= n; cnt++) {
1104             double x = a - f(a) * (a - b) / (f(a) - f(b));
1105             fprintf(fout, "%d. x = %.*lf\n", cnt,digit_count, x);
1106             if (fabs(x - x_old) <= eps) {
1107                 fprintf(fout, "Dung sau %d lan lap vi |xn - xn-1| <= %.9lf\n
    ", cnt, eps);
1108                 break;
1109             }
1110 
1111             a = x;
1112             double fa = f(a);
1113             double fb = f(b);
1114         }
1115         fclose(fout);
1116     }
1117 }
1118      // option 6
1119     void cau6(){
1120     system("cls");
1121     giao_dien_ngoai();
1122     giao_dien_ben_trong();
1123     giao_dienhuongdan();
1124 
1125     TextColor(252);
1126     gotoxy(8,8);printf("De chay chuong trinh ban su dung cach sau:");
1127     gotoxy(9,9);printf("De di chuyen len tren dung : mui ten xuong,s,S,2");
1128     gotoxy(9,10);printf("De di chuyen len tren dung: mui ten len,w,W,8");
1129     gotoxy(9,11);printf("De chon phan muon lam chon enter.");
1130     gotoxy(9,12);printf("Cau 3,5 chon lua chon 1 de chay");
1131     gotoxy(9,13);printf("Cau 3,5 chon 1 de kiem tra dieu kien");
1132     gotoxy(9,14);printf("Cau 4 chon lua chon 1 or 2 de chay");
1133     gotoxy(9,15);printf("Cau 4 lua chon 1,2 co chuc nang");
1134     gotoxy(9,16);printf("1.CT sai so:|f(x)|/m");
1135     gotoxy(9,17);printf("2.CT sai so (M-m)*(xn-xn-1)/m");
1136    }
```

```
1137
1138
1139
1140
1141
1142 void menu()
1143 {
1144   int position = 1;
1145     int keyPress = 0;
1146     giao_dien_nen();
1147     giao_dien_ben_trong();
1148     while (keyPress != 13) {
1149         for (int i = 1; i <= MAX; i++) {
1150             gotoxy(8, 9 + i);
1151             arrowHere(i, position);
1152             printf("%d. %s \n", i, option[i - 1]);
1153         }
1154         fflush(stdin);
1155         keyPress = _getch();
1156         if (keyPress == 80 || keyPress == 's' || keyPress == 'S' || keyPress
     == '2') {
1157             if (position == MAX) position = MIN;
1158             else position++;
1159         }
1160         else if (keyPress == 72 || keyPress == 'w' || keyPress == 'W' ||
    keyPress == '8') {
1161             if (position == MIN ) position = MAX;
1162             else position--;
1163         }
1164
1165 }
1166     switch (position) {
1167     case 1:
1168         cau1();
1169         _getch(); fflush(stdin);
1170         system("cls");
1171         menu();
1172         break;
1173     case 2:
1174         cau2();
1175         _getch();
1176         fflush(stdin);
1177         system("cls");
1178         menu();
1179         break;
1180     case 3:
1181         cau3();
1182         _getch();
1183         fflush(stdin);
1184         system("cls");
1185         menu();
1186         break;
1187     case 4:
1188       cau4();
1189         _getch();
1190         fflush(stdin);
1191         system("cls");
1192         menu();
1193         break;
1194     case 5:
```

```
1195        cau5();
1196         _getch();
1197         fflush(stdin);
1198         system("cls");
1199         menu();
1200         break;
1201     case 6:
1202        cau6();
1203         _getch();
1204   giao_dienhuongdan();
1205         fflush(stdin);
1206         system("cls");
1207         menu();
1208         break;
1209     case 7:
1210     exit(0);
1211     }
1212 }
1213
1214
1215 int main()
1216 {
1217   int n, stt = 1, choose;
1218   double a, b, eps;
1219   giao_dien_nen();
1220     enter_poly();
1221   menu();
1222     return 0;
1223 }
```

4 Hình ảnh giao diện thực hiện chương trình

4.1 Trường hợp nhập đúng, chính xác

Dưới đây là giao diện của chương trình và được nhập một cách tuần tự và chính xác. Xét trường hợp:

Phương trình đa thức f(x) bậc 4 và có dạng là: $f(x) = x^4 - 5x^3 + 3x^2 + 5x - 3$.

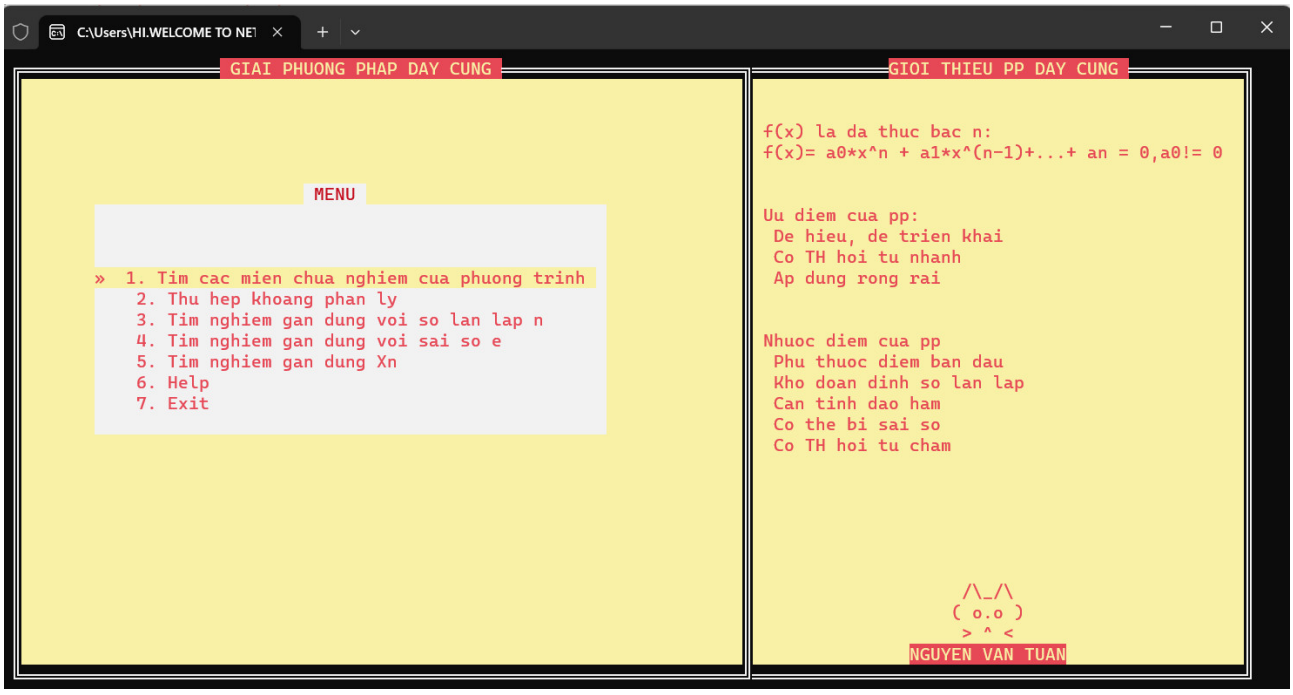
Trước tiên là giao diện đầu vào của chương trình, ở đây màn hình được chia làm 2 phần là một bên là nhập đầu vào và một bên là giới thiệu phương pháp dây cung với ưu nhược điểm của nó.

Hình 1: Giao diện nhập bậc và hệ số cho đa thức f(x)

Dưới đây là giao diện sau khi nhập bậc, hệ số và hiển thị kết quả sau dấu phẩy mấy số (được sử dụng để hiện kết quả xuyên suốt chương trình):

Hình 2: Giao diện sau khi nhập bậc và hệ số

Đây là giao diện sau khi nhập xong bậc, hệ số, giao diện này hiển thị các option để bạn lựa chọn gồm: 7 option dùng các nút trên bàn phím để di chuyển và tuỳ chọn option.



Hình 3: Giao diện menu

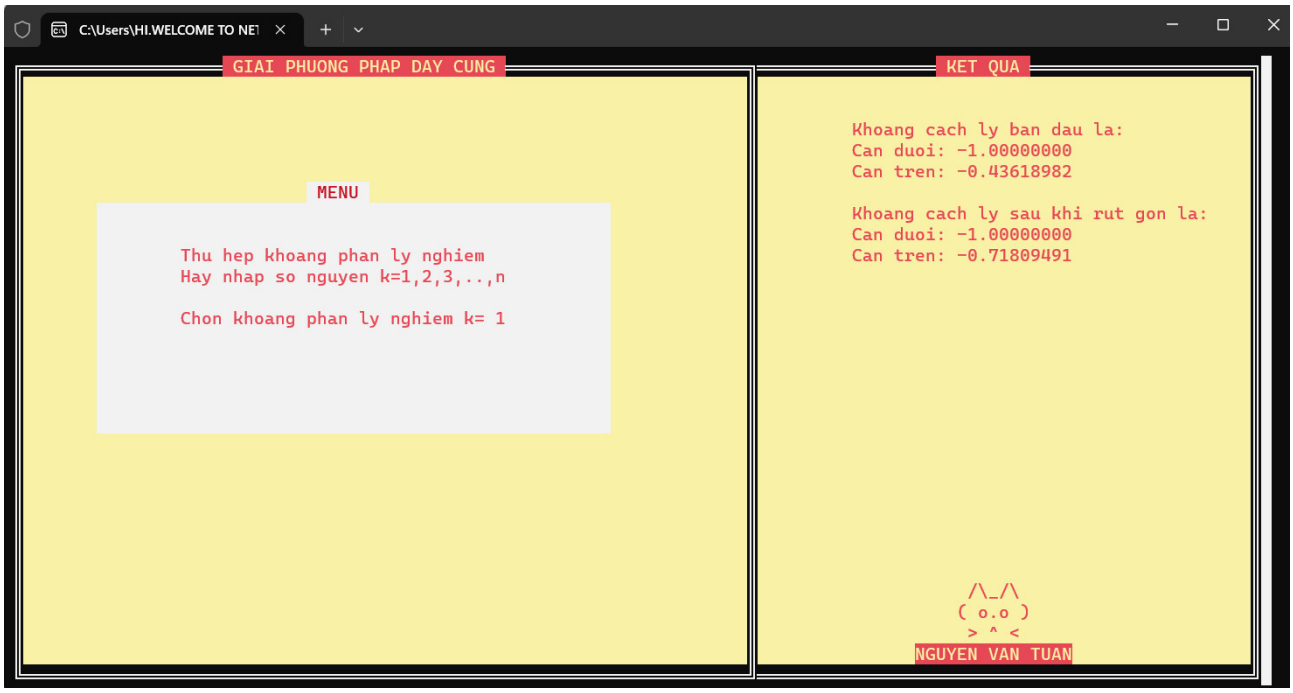
Kết quả của tìm kếm các miền chứa nghiệm và đưa ra thứ tự k=1,2,3... để bạn dùng trong option 2 và bên phải là những lưu ý để bạn nhập một cách chính xác nhất. Và 4 miền chứa nghiệm này chính xác và đã được xác thực.

Hình 4: Giao diện kết quả của câu 1

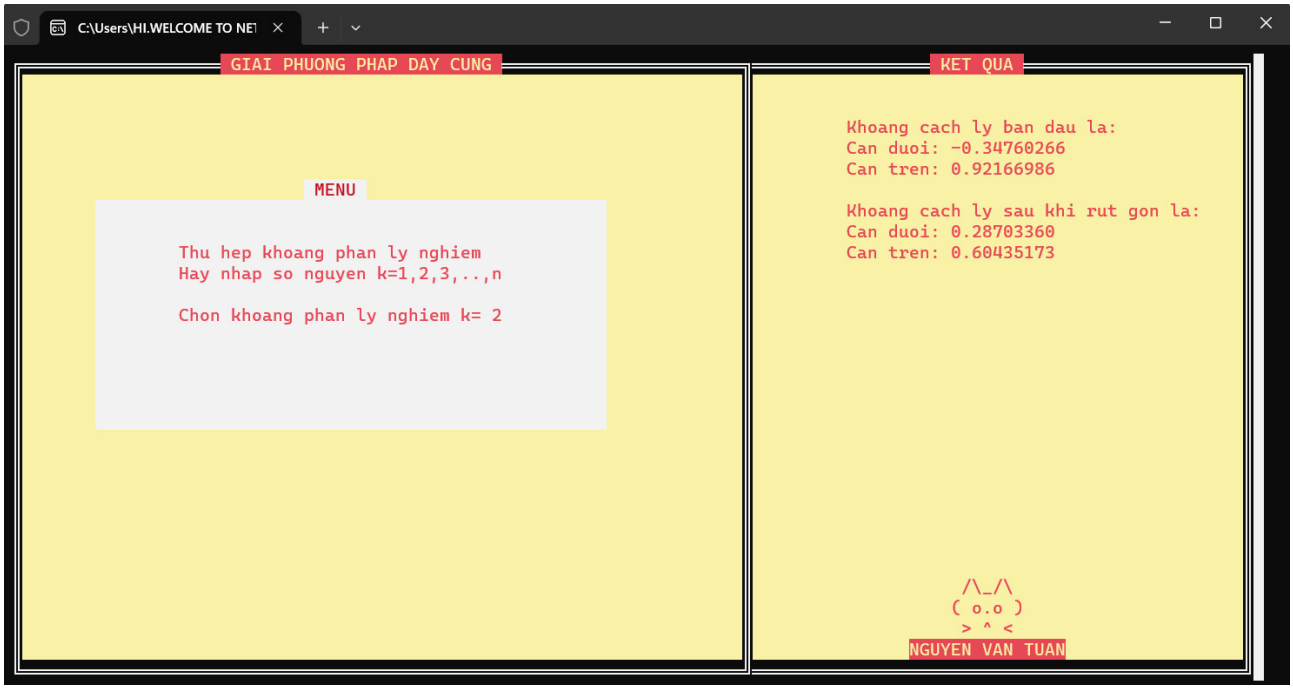
Giao diện của option 2, bạn sẽ sử dụng kết quả của option 1 để sử dụng.

Hình 5: Giao diện nhập của câu 2

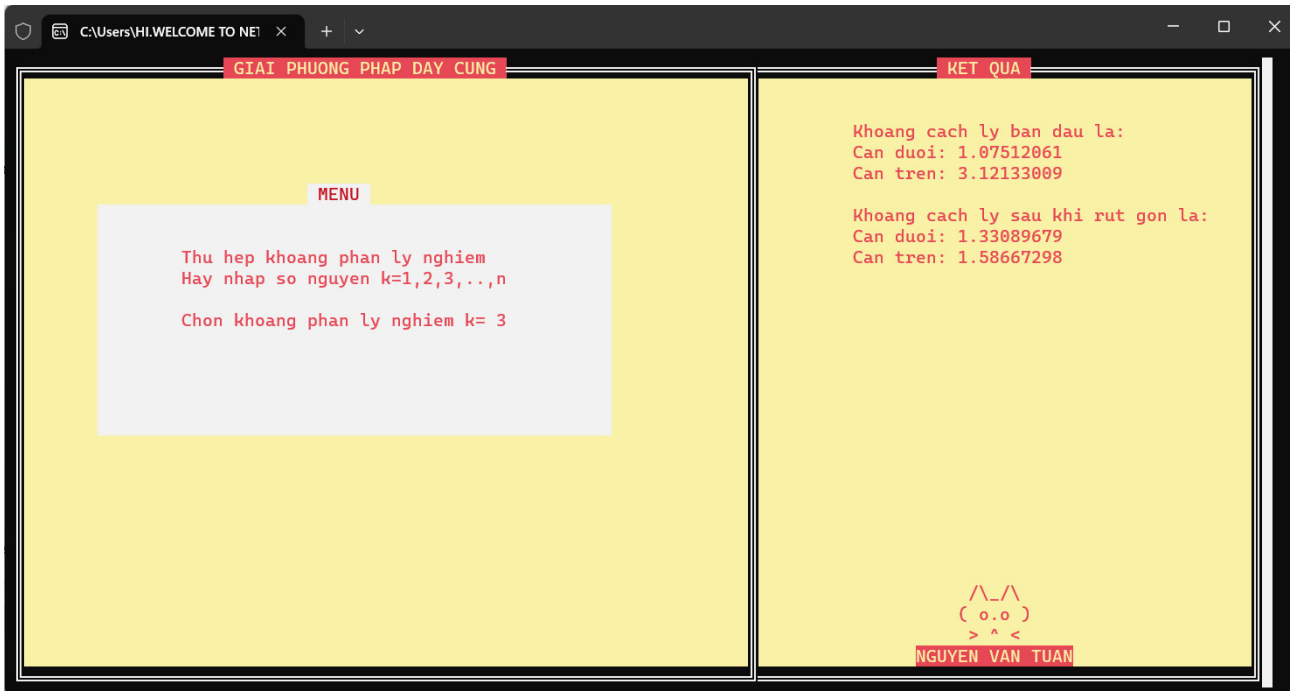
Dưới đây là kết quả thu hẹp 4 miền chứa nghiệm với |a-b|<=0.5



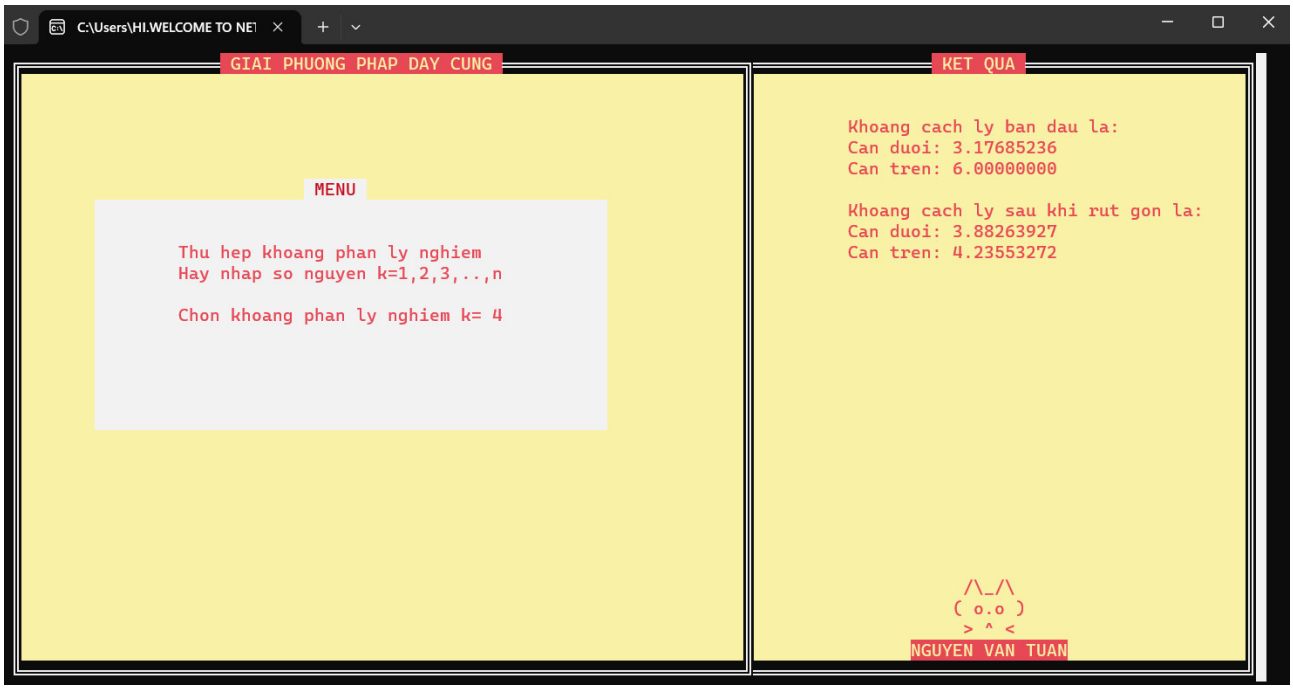
Hình 6: Giao diện kết quả của câu 2 khoảng phân ly 1



Hình 7: Giao diện kết quả của câu 2 khoảng phân ly 2

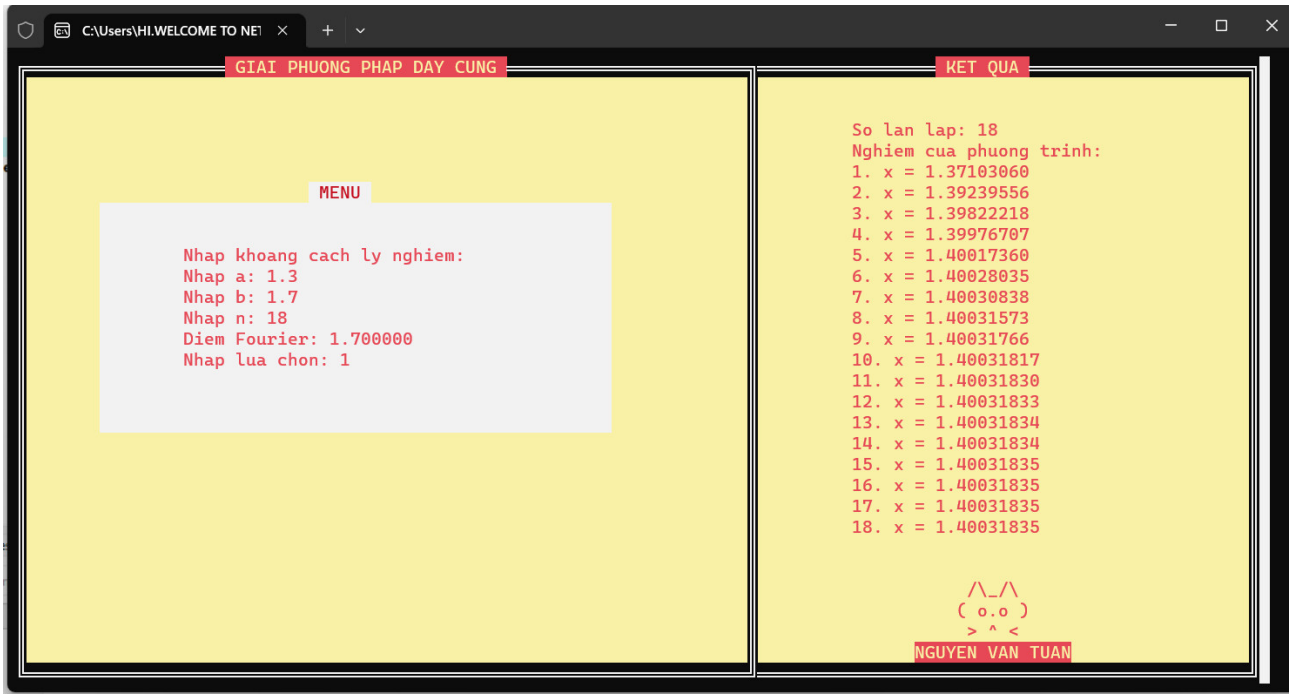


Hình 8: Giao diện kết quả của câu 2 khoảng phân ly 3



Hình 9: Giao diện kết quả của câu 2 khoảng phân ly 4

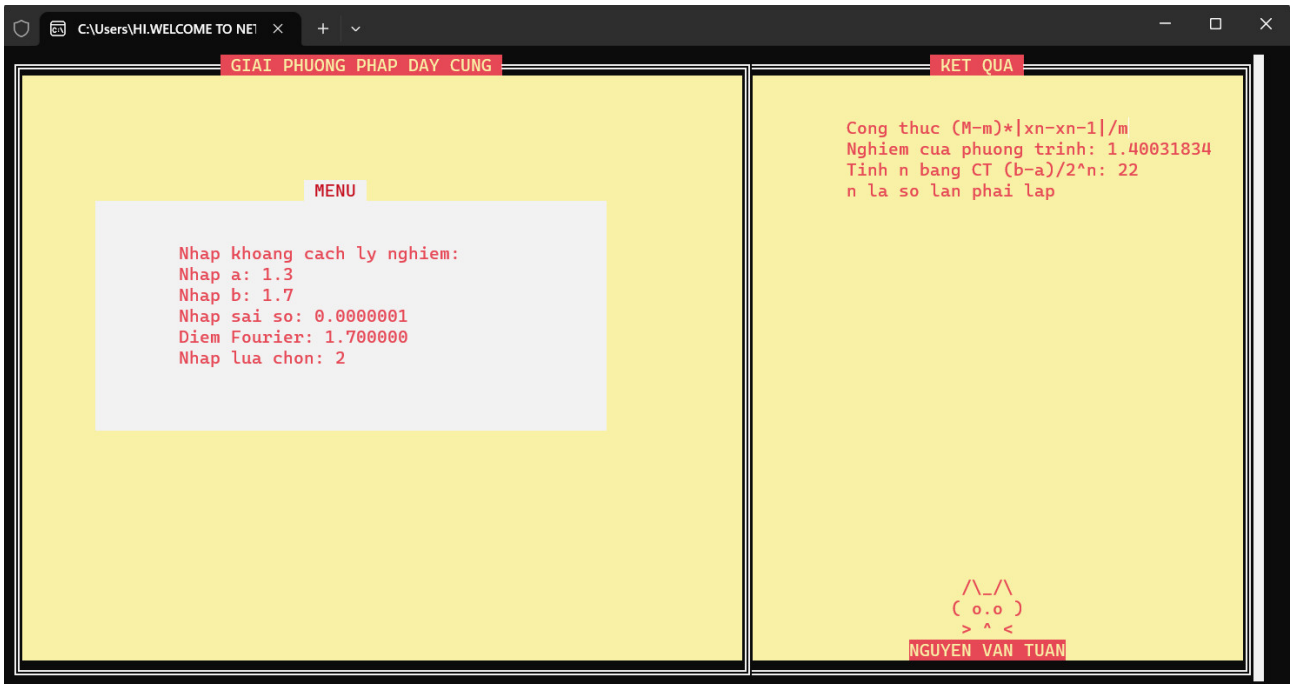
Giao diện kết quả option 3 với bên trái là khoảng ly nghiệm mà bạn muốn sử dụng để tìm nghiệm gần đúng sau n lần lặp (n được nhập từ bàn phím). Kết quả được hiển thị bên phải và điểm Fourier được hiển thị bên trái.



Hình 10: Giao diện kết quả câu 3

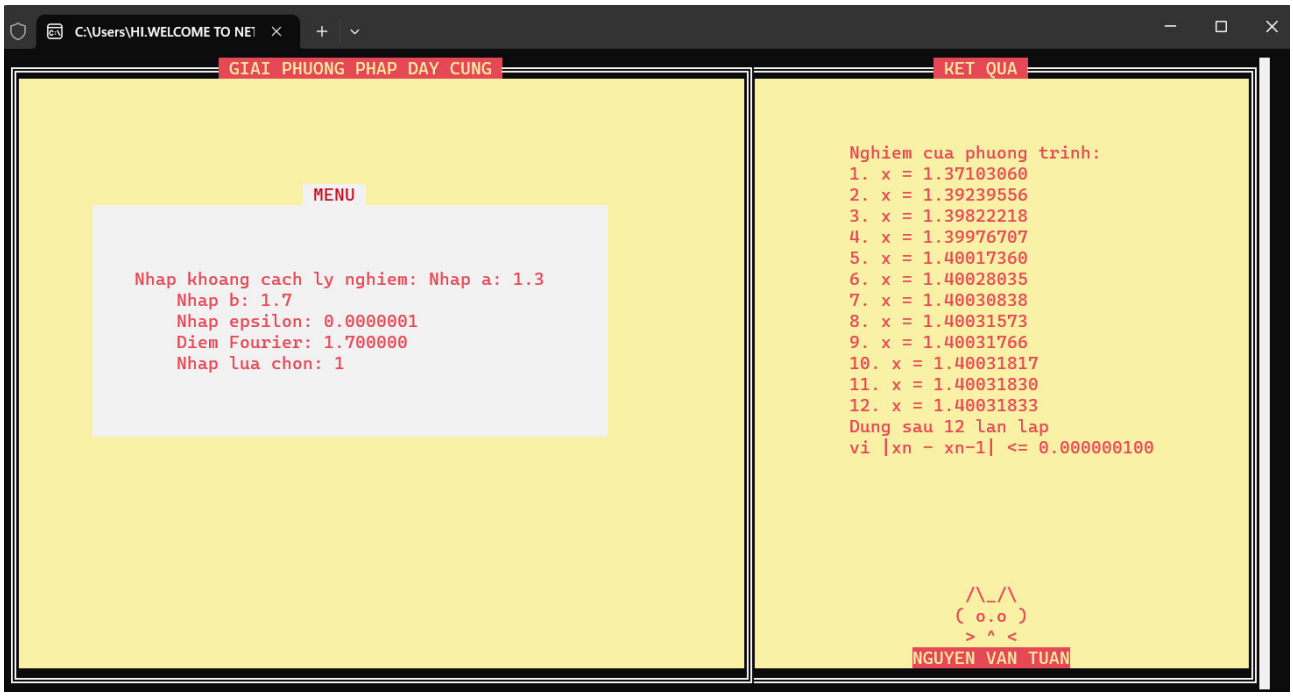
Giao diện kết quả option 4 với bên trái là khoảng ly nghiệm mà bạn muốn sử dụng để tìm nghiệm gần đúng với sai số e bạn muốn. Điểm Fourier được hiển thị bên tay trái và lựa chọn 1 hoặc để chọn công thức sai số bạn muốn thu gọn.

Hình 11: Giao diện kết quả của câu 4 với công thức sai số 1



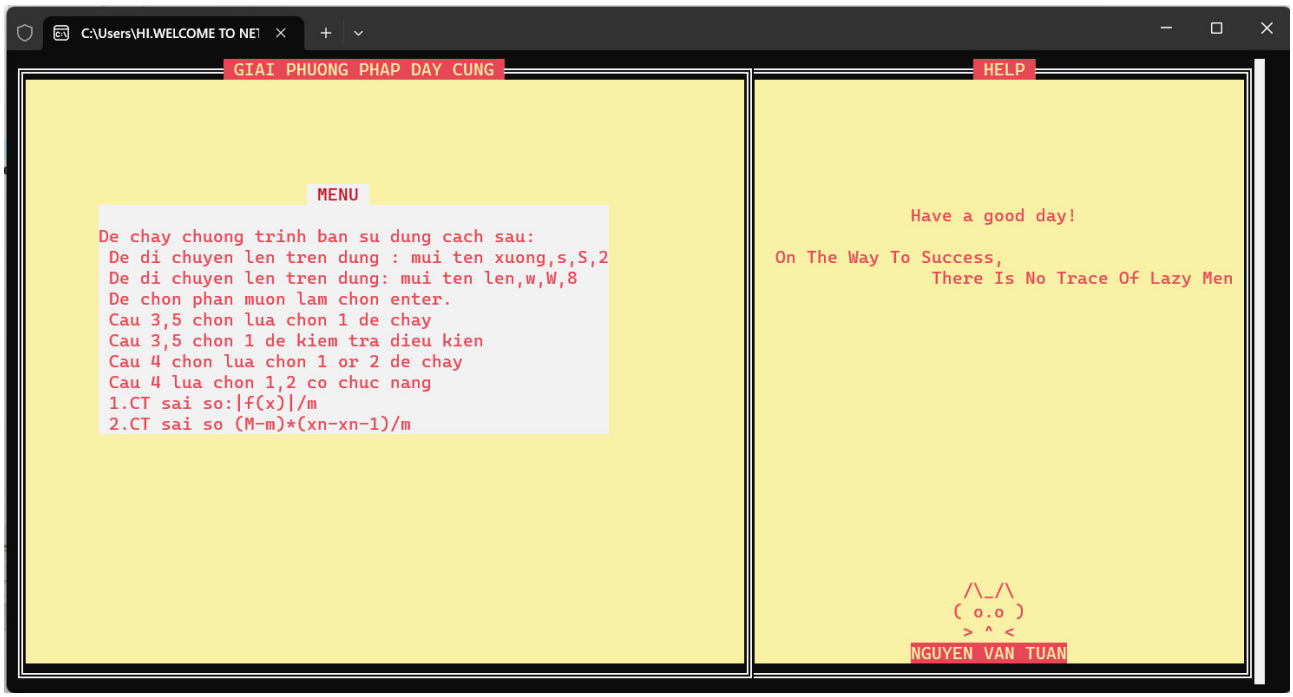
Hình 12: Giao diện kết quả của câu 4 với công thức sai số 2

Giao diện kết quả option 5 với bên trái là khoảng ly nghiệm mà bạn muốn sử dụng để tìm nghiệm gần đúng với điều kiện |Xn-Xn-1|<=e với e nhập từ bàn phím. Kết quả sẽ hiển thị số lần lặp đệ thoả mãn điều kiện trên.



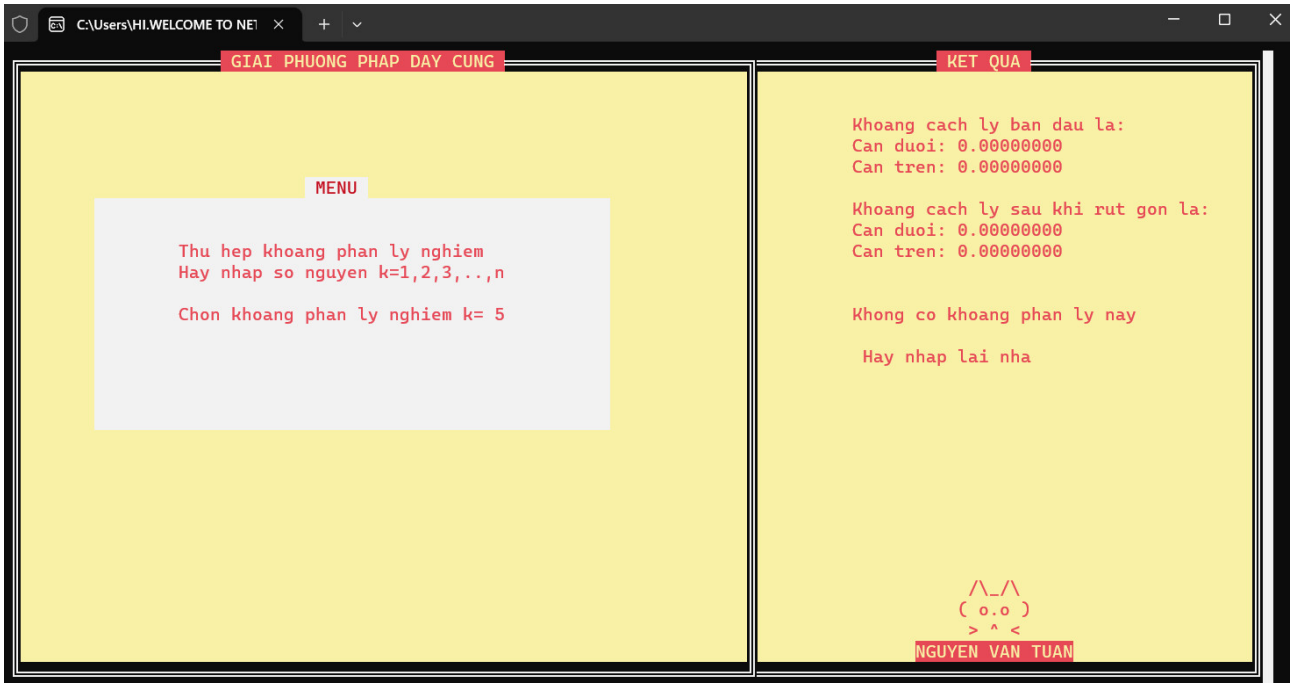
Hình 13: Giao diện kết quả của câu 5

Giao diện kết quả option 6 giúp bạn điều khiển chương trình một cách chính xác.



Hình 14: Giao diện hướng dẫn

4.2 Trường hợp nhập sai, lỗi



Hình 15: Giao diện kết quả option 2 với khoảng phân ly không tồn tại

Hình 16: Giao diện khi nhập không chính xác

Hình 17: Giao diện khi nhập không chính xác

Hình 18: Giao diện khi nhập không chính xác

Xét phương trình $f(x) = x^4 + 2x^3 + 2x^2 + 3x + 3$

Hình 19: Giao diện kết quả với trường hợp f(x) vô nghiệm

5 Kết quả của chương trình

```
Ban muon hien thi may so sau dau phay: 8
Bac cua da thuc: 4
Cac he so cua da thuc:
He so a0: 1.00000000
He so a1: -5.00000000
He so a2: 3.00000000
He so a3: 5.00000000
He so a4: -3.00000000

-----------------------
Cau 1: Tim mien chua nghiem cua phuong trinh da thuc fx
fx nam trong mat tron tam O, R: 6.00000000
Mien chua nghiem la:
Can duoi : -1.00000000
Can tren : 6.00000000
Khoang cach ly thu 1

Can duoi: -1.00000000
Can tren: -0.43618982

Khoang cach ly thu 2

Can duoi: -0.34760266
Can tren: 0.92166986

Khoang cach ly thu 3

Can duoi: 1.07512061
Can tren: 3.12133009

Khoang cach ly thu 4

Can duoi: 3.17685236
Can tren: 6.00000000

-----------------------
Cau 2: Thu hep khoang phan ly
Khoang cach ly ban dau:

Can duoi: -1.00000000
Can tren: -0.43618982

Lan lap thu 1

Can duoi: -1.00000000
Can tren: -0.71809491

-----------------------
Cau 2: Thu hep khoang phan ly
Khoang cach ly ban dau:

Can duoi: -0.34760266
Can tren: 0.92166986

Lan lap thu 1

```

```
Can duoi: 0.28703360
Can tren: 0.92166986

Lan lap thu 2

Can duoi: 0.28703360
Can tren: 0.60435173

-----------------------
Cau 2: Thu hep khoang phan ly
Khoang cach ly ban dau:

Can duoi: 1.07512061
Can tren: 3.12133009

Lan lap thu 1

Can duoi: 1.07512061
Can tren: 2.09822535

Lan lap thu 2

Can duoi: 1.07512061
Can tren: 1.58667298

Lan lap thu 3

Can duoi: 1.33089679
Can tren: 1.58667298

-----------------------
Cau 2: Thu hep khoang phan ly
Khoang cach ly ban dau:

Can duoi: 3.17685236
Can tren: 6.00000000

Lan lap thu 1

Can duoi: 3.17685236
Can tren: 4.58842618

Lan lap thu 2

Can duoi: 3.88263927
Can tren: 4.58842618

Lan lap thu 3

Can duoi: 3.88263927
Can tren: 4.23553272

-----------------------
Cau 2: Thu hep khoang phan ly
Khoang cach ly ban dau:

Can duoi: 0.00000000
Can tren: 0.00000000
Khong co khoang phan ly nay
-----------------------
```

```
Cau 3: Tim nghiem gan dung voi so lan lap n
Nghiem cua phuong trinh la: 1.400318
 nhap a: 1.30000000
 nhap b: 1.70000000
 nhap n: 18
 Diem Fourier: 1.70000000

 So lan lap: 18
Nghiem cua phuong trinh:
1. x = 1.37103060
2. x = 1.39239556
3. x = 1.39822218
4. x = 1.39976707
5. x = 1.40017360
6. x = 1.40028035
7. x = 1.40030838
8. x = 1.40031573
9. x = 1.40031766
10. x = 1.40031817
11. x = 1.40031830
12. x = 1.40031833
13. x = 1.40031834
14. x = 1.40031834
15. x = 1.40031835
16. x = 1.40031835
17. x = 1.40031835
18. x = 1.40031835

-----------------------
Cau 4: Tim nghiem gan dung voi sai so e
Nghiem cua phuong trinh la: 1.40031834
So lan lap: 1
 nhap a: 1.30000000
 nhap b: 1.70000000
 nhap sai so: 0.00000010
 Diem Fourier: 1.700000

 nhap lua chon: 1
Cong thuc |f(x)|/m
Nghiem cua phuong trinh la: 1.40031834
 Tinh n bang CT (b-a)/2^n: 22
 n la so lan phai lap
-----------------------
Cau 4: Tim nghiem gan dung voi sai so e
Nghiem cua phuong trinh la: 1.40031834
So lan lap: 1
nhap a: 1.30000000
 nhap b: 1.70000000
 nhap sai so: 0.00000010
 Diem Fourier: 1.700000

 nhap lua chon: 2
Cong thuc (M-m)*|xn-xn-1|/m
Nghiem cua phuong trinh la: 1.400318
 Tinh n bang CT (b-a)/2^n: 22
 n la so lan phai lap
 nhap lua chon: 2
-----------------------
Cau 5: Tm nghiem gan dung cua Xn
Nghiem cua phuong trinh la: 1.40031834
```

```
 nhap a:1.30000000
 nhap b: 1.70000000
 nhap epsilon: 0.00000010
 Diem Fourier: 1.70000000

Nghiem cua phuong trinh:
1. x = 1.37103060
2. x = 1.39239556
3. x = 1.39822218
4. x = 1.39976707
5. x = 1.40017360
6. x = 1.40028035
7. x = 1.40030838
8. x = 1.40031573
9. x = 1.40031766
10. x = 1.40031817
11. x = 1.40031830
12. x = 1.40031833
13. x = 1.40031834
Dung sau 12 lan vi |xn-xn-1|<= 0.000000100
```

6 Đánh giá chung

BẢNG ĐÁNH GIÁ CHUNG MÔN HỌC: KỸ THUẬT LẬP TRÌNH				
NGUYỄN VĂN TUẤN				
STT	**Nội dung yêu cầu (theo đề)**	**Đã viết code? (Y/N)**	**Đã thực hiện đúng? (Y/N)**	**Tự đánh giá nhược điểm, ưu điểm, những sáng tạo**
1	Tìm các miền chứa nghiệm của phương trình	Y	Y	Đã tìm được đầy đủ các miền nghiệm
2	Tìm khoảng phân ly nghiệm (a, b) của phương trình thoả mãn $\|a-b\| \leq 0.5$ bằng cách sử dụng phương pháp chia đôi để thu hẹp dần một khoảng phân ly nghiệm đã tìm được ở ý 1)	Y	Y	Đã thu hẹp các khoảng phân ly thoả mãn điều kiện
3	Tìm nghiệm gần đúng với số lần lặp n cho trước trong khoảng phân ly nghiệm (a, b) và đánh giá sai số theo cả hai công thức (n được nhập vào từ bàn phím, (a, b) được nhập vào từ bàn phím)	Y	Y	Đã hoàn thành đầy đủ
4	Tìm nghiệm gần đúng trong khoảng (a, b) với sai số e cho trước (e được nhập vào từ bàn phím, (a, b) được nhập vào từ bàn phím). Tính toán theo 2 cách áp dụng công thức sai số	Y	Y	Đã hoàn thành đầy đủ
5	Tìm nghiệm gần đúng x_n trong khoảng (a, b) thoả mãn điều kiện: $\|x_n - x_{n-1}\| \leq e$ (e được nhập vào từ bàn phím)	Y	Y	Đã hoàn thành đầy đủ
6	Mọi kết quả được hiển thị với số chữ số phần thập phân nhận vào từ bàn phím	Y	Y	Đã hoàn thành đầy đủ
7	Ghi vào tệp văn bản thể hiện quá trình thực hiện chương trình và các kết quả ra	Y	Y	Đã hoàn thành đầy đủ
8	Thực hiện chương trình bằng menu điều khiển bởi các phím chức năng. Sinh viên tự code để thiết lập và điều khiển menu	Y	Y	Đã hoàn thành đầy đủ

Bảng 1: Đánh giá chung

7 Tổng kết

Sau khi hoàn thành đề bài chủ đề 3, bản thân em đã đạt được những điều sau:

- Hiểu hơn về vai trò cũng như lợi ích của ngôn ngữ C, cũng như ngôn ngữ lập trình khác.
- Hiểu về cấu trúc để tạo ra một chương trình lớn hơn.
- Sử dụng thành thạo hơn ngôn ngữ C.
- Tự tìm tòi học những điều mới khi đi tìm ý tưởng để làm chương trình.
- Hiểu rõ hơn, được ôn tập luôn môn giải tích số, nhận thấy toán học chính là nền tảng để có kiến thức, tư duy lập trình tốt hơn.

Bên cạnh đó là những thiếu sót mà bản thân cần khắc phục:

- Kiến thức toán chưa được chắc chắn, mất nhiều thời gian để hiểu và viết ra chương trình.
- Giao diện chưa được bắt mắt, sáng tạo lắm.
- Code có đoạn vẫn hơi dài chưa được tối ưu triệt để.
- Và còn một số thiếu sót nhỏ,..

Và đến đây là kết thúc báo cáo của em, cảm ơn cô đã dạy cho em những điều bổ ích về học phần này, chúc cô luôn khoẻ mạnh, thành công trong công tác giảng dạy cũng như trong cuộc sống, mong lại được học cô ở những học phần tiếp theo.

Trân trọng cảm ơn