Cấu trúc dữ liệu và giải thuật

Báo cáo

I. Cấu trúc chương trình

1. Hàm liên quang đến thuật toán balan(Stack và Queue).
    - void KhoitaoStack(StackSothuc &stacksothuc, StackKitu &stackkitu)
    - NodeChuoi *TaoNodeChuoi(char *chuoi)
    - NodeKitu *TaoNodeKitu(char kitu)
    - NodeSothuc *TaoNodeSothuc(double sothuc)
    - void KhoitaoQueue(Queue &queue)
    - bool isEmptyChuoi(Queue queue)
    - bool isEmptyKitu(StackKitu stackkitu)
    - bool isEmptySothuc(StackSothuc stacksothuc)
    - bool PopKitu(StackKitu &stackkitu, char &kitu)
    - bool PopSothuc(StackSothuc &stacksothuc, double &sothuc)
    - char top(StackKitu stackkitu)
    - bool PushKitu(StackKitu &stackkitu, NodeKitu *node)
    - bool PushSothuc(StackSothuc &stacksothuc, NodeSothuc *node)
    - bool deQueue(Queue &queue, char chuoi[])
    - bool enQueue(Queue &queue, NodeChuoi *node)
    - bool KiemtraUuTien(char a, char b)
    - void Hauto(char bieuthuc[], Queue &queue, StackKitu &stackkitu)
2. Hàm sử lý tính toán.
    - double luythua(double numberone, double numbertwo)
    - double giaithua(double number)
    - double Tinhtoan(double numberone, double numbertwo, char pheptoan)
    - bool SuLytinhtoan(StackSothuc &stacksothuc, Queue &queue)

3. Hàm kiểm tra tính đúng đắng của dữ liệu vào và sử lý dử liệu.
    - void xoaKhoangTrang(char bieuthuc[])
    - bool SuLytinhtoan(StackSothuc &stacksothuc, Queue &queue)
    - bool kiemTraBieuThuc(char *bieuthuc)

II. **Công dụng của các hàm trong chương trình**.

1. Hàm **KhoitaoStack** có nhiệm vụ khởi tạo giá trị ban đầu cho stack.
2. Hàm **TaoNodeChuoi** có nhiệm vụ tạo một node để chứa dữ liệu là chuỗi.
3. Hàm **TaoNodeKitu** có nhiệm vụ tạo một node để chứa dữ liệu là kí tự.
4. Hàm **TaoNodeSothuc** có nhiệm vụ tạo một node để chứa dữ liệu là số thực.
5. Hàm **KhoitaoQueue** có nhiệm vụ khởi tạo giá trị ban đầu cho queue.
6. Hàm **isEmptyChuoi** kiểm tra xem Queue chứa chuỗi có rỗng hay không.
7. Hàm **isEmptyKitu** kiễm tra xem Stack chứa kí tự có rỗng hay không.
8. Hàm **isEmptySothuc** kiểm tra Stack chứa số thực có rỗng hay không.
9. Hàm **PopKitu** lấy dử liệu dạng kí tự từ Stack và xóa nó khỏi Stack.
10. Hàm **PopSothuc** lấy dử liệu dạng số thực từ Stack và xóa nó khỏi Stack.
11. Hàm **top** dùng để lấy dử liệu từ Stack nhưng không xóa nó khỏi Stack.
12. Hàm **PushKitu** dùng để thêm dử liệu dạng kí tự vào Stack kí tự.
13. Hàm **PushSothuc** dùng để thêm dử liệu dạng số thực vào Stack số thực.
14. Hàm **deQueue** dùng để xóa 1 phần tử trong Queue.
15. Hàm **enQueue** dùng để thêm 1 phần tử vào Queue.

16. Hàm **KiemtraUuTien** dùng để kiểm tra độ ưu tiên của các toán tử để thực hiện thuật toán balan.
17. Hàm **Hauto** dùng để chuyển biểu thức dạng trung tố về hậu tố.
18. Hàm **luythua** dùng để tính lũy thừa trong biểu thức.
19. Hàm **giaithua** dùng để tính giai thừa trong biểu thức.
20. Hàm **Tinhtoan** dùng để tính toán kết quả giữa 2 số được lấy từ Stack.
21. Hàm **SuLytinhtoan** dùng tính toán ra kết quả cuối cùng và củng dùng để kiểm tra độ đúng đắng của biểu thức.
22. Hàm **xoaKhoangTrang** dùng để xóa khoảng trắng trong biểu thức input.
23. Hàm **SuLytinhtoan** dùng để kiểm tra độ đúng đắng của phép tính giai thừa.
24. Hàm **kiemTraBieuThuc** dùng để kiểm tra độ đúng đắng của biểu thức, nếu đúng sẻ trả về true.