BÁO CÁO

a/ Viết hàm chuyển từ biểu thức trung tố về biểu thức hậu tố sử dụng thuật toan Ba Lan ngược.

Ý tưởng:

Nhập vào 1 chuỗi. Xét lần lượt x từ đầu đến cuối chuỗi. Nếu x:

TH1: x là toán hạng => xuất ra màn hình.

TH2: x là toán tử

TH2.1: stack không rỗng

Nếu độ ưu tiên của Top trong stack lớn hơn hoặc bằng độ ưu tiên của x thì xuất Top ra màn hình và pop stack.

TH2.2: Đỉnh stack không phải toán tử hoặc độ ưu tiên của phần tử Top nhỏ hơn độ ưu tiên của x thì push(x) và stack.

TH3: x là “(” => push(x).

TH4: x là “)” => xuất Top cho đến khi gặp “(”.

Thuật toán:

- Hàm tính độ ưu tiên toán tử

```
int UuTien(char op)
{
	if (op == '+' || op == '-')
		return 1;
	if (op == '*' || op == '/')
		return 2;
	return -1;
}
```

Hàm nhận vào 1 kí tự kiểu char và trả về độ ưu tiên của kí tự nếu là toán tử

- Hàm chuyển biểu thức trung tố về hậu tố

```
void infixtoPostfix(string a)
{
	Stack s;
	s.init();
```

```
	for (int i = 0; i < a.size(); i++)
	{
		if (a[i] >= '0' && a[i] <= '9')
		{
			cout << a[i] << " ";
		}
		else if (a[i] == '+' || a[i] == '-' || a[i] == '*' || a[i] == '/')
		{
			if (!s.isEmpty())
			{
				if (UuTien(s.Top()) >= UuTien(a[i]))
				{
					cout << s.Top() << " ";
					s.pop();
				}
			}
			if (s.isEmpty() || s.Top() == '(' || UuTien(s.Top()) <
UuTien(a[i]))
			{
				s.push(a[i]);
			}
		}
		else if (a[i] == '(')
		{
			s.push(a[i]);
		}
		else
		{
			while (s.Top() != '(')
			{
				cout << s.Top() << " ";
				s.pop();
			}
			s.pop();
		}
	}
	// Xuất hết stack
	while (!s.isEmpty()) {
		cout << s.Top() << " ";
```

```
            s.pop();
        }
}
```

Hàm nhận vào 1 chuỗi là biểu thức trung tố và sau khi chạy xong hàm xuất ra màn hình là biểu thức đã được chuyển đổi về hậu tố.

c/ Viết hàm tính toán biểu thức hậu tố

Ý tưởng:

Nhập vào 1 chuỗi. Xét lần lượt x từ đầu đến cuối chuỗi.

TH1: Nếu x là số thì chuyển x từ chuỗi sang số và push(x) vào stack.

TH2: Nếu x là toán tử thì lấy 2 giá trị từ đỉnh stack, thực hiện phép tính rồi push kết quả vào trong stack.

Xử lí đến cuối cùng thu được kết quả là phần tử ở đỉnh của stack.

Thuật toán:

```
int tinh_postfix(string ht)
{
        Stack s;
        s.init();

        for (int i = 0; i < ht.length(); i++)
        {
                if (ht[i] == ' ')
                {
                        continue;
                }
                if (isdigit(ht[i]))
                {
                        int num = 0;
                        while (isdigit(ht[i])) // chuyển chuỗi thành số
                        {
                                num = num * 10 + (ht[i] - '0');
                                i++;
                        }
                        i--;
                        s.push(num);
                }
                else
```

```
            {
                int a = s.Top();
                s.pop();
                int b = s.Top();
                s.pop();
                switch (ht[i])
                {
                case '+':
                    s.push(b + a);
                    break;
                case '-':
                    s.push(b - a);
                    break;
                case '*':
                    s.push(b * a);
                    break;
                default:
                    break;
                }
            }
        }
        return s.Top();
}
```

Hàm này dùng để tính giá trị biểu thức hậu tố.

Xét lần lượt chuỗi “ht”. Nếu

- ht[i] là kí tự trống thì bỏ qua và tiếp tực vòng lặp.
- ht[i] là kí tự số thì chuyển đổi ht[i] thành số và push vào stack.
- ht[i] là toán tử thì lấy ra 2 giá trị ở đỉnh stack và thực hiện phép tính tương ứng sau đó push kết quả vào stack.

Sau khi kết thúc vòng lặp thì phần tử ở đỉnh stack là kết quả cần tìm.