150. Evaluate Reverse Polish Notation

You are given an array of strings tokens that represents an arithmetic expression in a Reverse Polish Notation.

Evaluate the expression. Return *an integer that represents the value of the expression*.

Note that:

- The valid operators are '+', '-', '*', and '/'.
- Each operand may be an integer or another expression.
- The division between two integers always **truncates toward zero**.
- There will not be any division by zero.
- The input represents a valid arithmetic expression in a reverse polish notation.
- The answer and all the intermediate calculations can be represented in a **32-bit** integer.

Example 1:

Input: tokens = ["2","1","+","3","*"]

Output: 9

Explanation: ((2 + 1) * 3) = 9

Example 2:

Input: tokens = ["4","13","5","/","+"]

Output: 6

Explanation: (4 + (13 / 5)) = 6

Example 3:

Input: tokens = ["10","6","9","3","+","-11","*","/","*","17","+","5","+"]

Output: 22

Explanation: ((10 * (6 / ((9 + 3) * -11))) + 17) + 5

= ((10 * (6 / (12 * -11))) + 17) + 5

= ((10 * (6 / -132)) + 17) + 5

= ((10 * 0) + 17) + 5

= (0 + 17) + 5

= 17 + 5

= 22

```
class Solution {
public:
    int evalRPN(vector<string>& tokens) {
        int ans = 0;
        stack<int> st;
        for(auto ch : tokens){
            if(ch == "+" || ch == "-" || ch == "/" || ch == 
"*"){
                int a = st.top();
                st.pop();
                int b = st.top();
                st.pop();
                if (ch == "+")
                    st.push(b + a);
                else if (ch == "-")
                    st.push(b - a);
                 else if (ch == "*")
                    st.push(b * a);
                 else if (ch == "/")
                    st.push(b / a);

            }else{
                st.push(stoi(ch));
            }
        }
        return st.top();
    }
};
```