```
void listElements() {
        // if the stack is empty, print out that's its empty
        if (isEmptyStack()) cout << "The stack is empty";
        // else iterate through the stack and print out all its elements
        else {
                for (int i = stackTop-1; i >= 0; i--) {
                        cout << list[i] << endl;
                }
        }
}

// Function that changes the third value in a stack
void changeThirdValue(Type t) {
        // If the stack does not have enough elements, print out it does not have have enough elements
        if (stackTop < 3) cout << "Not enough elements";
        // Else iterate through the stack. When the third element in the stack is reached, replace it with t
        else {
                for (int i = stackTop - 1; i >= 0; i--) {
                        if (i == 3) list[i-1] = t;
                }
        }
}
```

In order to list all the elements in a stack as well as change its values, there must be a way to iterate through the stack. The way I implemented an iterator is with a simple for loop. Since the data structure is a stack, the stack should iterate from the last element to the first element. For the loop I set i to stackTop - 1 which represents the top position of the stack and i is decremented until i is 0 which represents that it has reached the first element.

For the function that lists all the elements of the stack, the function starts from the top of the stack and prints out each element as it makes its way to the bottom of the stack. For the function that changes the a value, for example the third value of the stack, the for loop iterates through the stack until the third value from the top is reached, and there the value is changed to whatever the user input.