```
void Stack::push(int _value)
{
  StackNode* temp = new StackNode();  1
  temp->data = _value;  1
   if (isEmpty())  1
   {
          temp->next = tail; 1
          head = temp;  1
          tail = temp;  1
   }
   else  0
   {
      tail->next = temp;  1
      tail = temp;  1
   }
}
```

With if condition:

2 + 4 = 6 steps

With else condition:

2 + 3 = 5 steps

MAX : 6

Repetition : n

6*n = 6n → O(n)

n = count of members

```
int Stack::pop()
{
if (isEmpty())  1
   {
      cout << "Stack is empty!!!" << endl;  1
      exit(1);  1
   }
   else  0
   {
      StackNode* temp;  0
      temp = head;  1
      int value ;  0
      if (head == tail)  1
      {
         value = head->data;  1
         delete head;  1
         head = NULL;  1
          tail = NULL;  1
           return value;  1
       }
      while (temp->next != tail )  n (one more time than n-1)
            temp = temp->next;       n-1 ( from head(member 0 )to member before tail(member n-2))
```

```
 tail = temp; 1
temp = temp->next;  1
tail->next = NULL ;  1
value= temp->data;  1
delete temp;  1
return value;  1
   }
}
```

With if condition:

3 steps

With else condition:

With inner if condition:

2 + 6 = 8 steps

Without inner if condition:

2 + 2n + 6 = 2n + 8 steps

MAX : 2n + 8

Repetition : n

(2n +8) * n = 2n^2 + 8n → O(n^2)