Omar Gonzalez
Dr. Guillen
CS 421
Project 2
Purpose: To implement a parser that processes sequences of correctly paired ifs and elses in a C++ program. by simulating a PDA

P = ({q0,q1,q2,q3,qf,qe},{if,else,{,}}, {a,b,z},δ,q0,z,{qf})
qe is the error state.
δ is define as follows:
1) δ(q0, λ , z) = {(qf, z)}
2) δ(q1, if , z) = {(q1, az)}
3) δ(q1, if , a) = {(q1, aa)}
4) δ(q1, else , z) = {(qe, z)}
5) δ(q1, else , a) = {(q1, λ)}
6) δ(q1, } , a) = {(qe, a)}
7) δ(q1, { , z) = {(q2, z)}
8) δ(q1, { , a) = {(q2, ba)}
9) δ(q2, { , b) = {(q2, bb)}
10) δ(q2, } , b) = {(q1, z)}
11) δ(q2, if , b) = {(q3, ab)}
12) δ(q3, if , a) = {(q3, aa)}
13) δ(q3, else , a) = {(q3, λ)}
14) δ(q3, } , a) = {(q2, λ)}
15) δ(q3, } , a) = {(q2, λ)}
16) δ(q2, λ , z) = {(qf, λ)}
17) δ(q2, λ , a) = {(qf, a)}
18) δ(q1, λ , a) = {(qf, a)}
19) δ(q2, λ , b) = {(q1, λ)}
20) δ(q1, λ , z) = {(qf, z)}
21) δ(q0, else , z) = {(qe, z)}
22) δ(q0, { , z) = {(qe, z)}

CFG:
G = ({S,A}, {if,else,{,}}, S, P),
where P is define as follows:
S -> ifA | ifAelse | ifelseA | λ
A -> {A} | S | λ

Description:
For my Push-Down Automata, if your read anything else other than an If or empty string in q0 it goes to the error the state(qe). If it reads an ‘if’ in q0 and it already has a ‘z’ on top of the stack, then it pushes an ‘a’ on top of the stack and moves from q0 to q1. At q1 I had other things to take into consideration but some of them are the following. If it reads more ‘If’s it will keep pushing a’s, if it read ‘else’s it pops until there are no more ‘a’s to pop or is the empty string. Once/if It reads opening bracket in state 1(q1) then it pushes a ‘b’ into our stack and moves to a different state(q2). If it reads its first ‘if’ in q2 and there is a ‘b’ on top of the stack it moves to q3. if at q2 it reads an else and there is a ‘b’ on top of the stack it will go to our error state(qe). when it reads an else’s in q3 it will pop an ‘a’ until there is a ‘b’ on top of the stack or it reads the closing brackets’}’. but if it read ‘if’s in q3 It will keep pushing ‘a’s and stay in the same state until I read my first closing bracket ‘}’. Then this will make my PDA to pop and move back to q1, to then repeat all the same steps until I reach my finals state.