```
///////////////////////////////////////////////////////////////////////////////
////Class:        CS 445
////Semester:     Fall 2011
////Assignment:   Homework 4
////Author:       Colby Blair
////File name:    tree_symtab_gen.h
////////////////////////////////////////////////////////////////////////////////

#include "tree.h"
#include "symtabc.h"
#include "tree_symtab_gen.h"
#include "parser.tab.h"
#include "main.h"


int tree_gen_assignmentExpression(struct tree *t)
{
        //get the token name
        char *tname = tree_get_ident(t);

        //if no token name, this is a right hand assignmentExpression with a
        // literal, which we'll ignore now and handle when we see the entire
        //expression later (below)
        if(tname == NULL)
        {
                return(0);
        }
        //else, this is an entire assignmentExpression to a identifier
        else
        {
                SymTabEntry *p = SymTab_lookup(tname);

                //get the subtree symbol for report info
                struct tree *temp = NULL;
                tree_get_subtree(tname, t, &temp);

                //use before declaration check
                if(p == NULL)
                {
                        if(temp != NULL)
                        {
                                struct tree_token *symbol = temp->leaf;
                                fprintf(stderr,
                                "ERROR: identifier '%s' used before declared, in fil
e '%s' on line %d\n",
                                        tname, symbol->fname, symbol->lineno);
                        }
                        else
                        {
                                fprintf(stderr,
                                "ERROR: identifier '%s' used before declared, and so
me horrible internal errors also occured.\n",
                                        tname);
                        }

                        exit(ERROR_SEMANTIC);
                }

                //assignment to const check
                if(p->aux_flag != NULL
                        && strcmp(p->aux_flag, "const") == 0)
                {
                        if(temp != NULL)
                        {
                                struct tree_token *symbol = temp->leaf;
                                fprintf(stderr,
                                 "ERROR: assignment to constant '%s', in file '%s' o
n line %d\n",
                                tname, symbol->fname, symbol->lineno);

                        }
                        else
                        {
                                fprintf(stderr,
                                 "ERROR: assignment to constant '%s', and some horrible internal errors also occured.\n",
                                 tname);
                        }

                        exit(ERROR_SEMANTIC);
                }
        } //end else (if tname != NULL

        //otherwise, some tac gen here

        return(0); //success
}
```