学籍番号、744177

氏名、上田　晃

グループ、G7

担当したソースコード

exp_imp.h

expnode.h

expression.c

expression.h

exptree.c

functions.c

丁寧に書いてあって良いと思いますが、
基本的な部分を少し理解できていない感じがします。
仕事に関係なく、C言語を勉強できる最後の機会かも
しれません。頑張りましょう。

行間が妙に開いていますが、もう少し詰めてもいいと思います。

・exp_imp.h

2~3 もし_EXP_IMP_H__が定義されていないなら定義する。

＃if !defined(__EXP_IMP_H__)から、＃endif で囲まれた部分はインクルードガ

ードという手法を使っており、exp_imp.h が初めてインクルードされた時は

#define 部分が未定義であるため#if !defined と#endif で囲まれた部分がプログラムとして読み込まれて定義される。しかし、2 回目以降のインクルードでは #define 部分が定義済みであるため読み飛ばされるようになる。

struct _expnode、struct _oprExpmode、struct _argExpnode では、token.h で宣言されている token 型の tok_id、tok_num、tok_str、などの kind と prefix はそれぞれ共通したものとなっている。

struct _expnode では union という構造体とよく似た共用体と呼ばれるものでありそこには long 型の intvalue、var の情報を入れる varinfo が定義されている。struct _oprExpmode では*operand[2]の配列が、struct _argExpnode では short 型の index,count と*args[1]の配列が定義されている。

３つの構造体に共通部分があることと、struct _argExpnode に
長さ１の配列があることを、少し気に留めておいて下さい。

・expnode.h

インクルードガードの手法を用いられている。

5~10 では、exp_imp.h で定義した struct _expnode、struct _oprExpmode、struct _argExpnode を呼び出ししている。

関数ではないので「呼び出す」という説明はおかしい。

expnode *newExpnode では_expnode の kind と prefix を呼び出す。

expnode *newOprnode では_oprExpmode の kind と*operand[2]を呼び出す。

argExpnode *newArgnode では_argExpnode の prefix と short 型の index と count を呼び出す。

間違っています。
ここはプロトタイプ宣言ですよ。

・expression.c

static int precedence(token_t op)では二項演算子を表すトークンを引数として、演算子の優先順位を返す。

そうですね！

構造体 oppbody には、 配列 opstack に演算子、配列 prec にはその演算子の優先順位、配列 nodestack には項を表す木へのポインタを格納する。nodindex に

はその配列の末尾＋1 を格納する。

関数 oppPutOperator()は関数 expression()から呼び出されるものである。nodestack に term()が格納されてその対応する二項演算子とともに関数 oppPutOperator()が呼び出される。

関数 expressionList では、配列型の xlist と int 型の args を引数としている。s を関数 getItem()で取得後、args が 0 より大きければ xlist の配列を args の長さ分を関数 expression()で取得する。

つまりこの関数の機能は何だろうか？

・expression.h

インクルードガードの手法を用いている。

expression.c の expression(void)、strexpression(void)、expressionList(expnode xList[]、int args)を呼び出す。

間違っています。
ここはプロトタイプ宣言です。

・exptree.c

関数 newExpnode()ではプログラム中に診断機能を付け加える assert を宣言しており、assert が実行された時、つまり kind が tok_id または tok_num または tok_str と同じ値の時にプログラムを中止する。malloc(size)とは size 分のメモリ領域を確保する関数である。xp の構造体のデータを引数に代入して、xp に戻り値を返す。

逆です。定義を参照のこと。

関数 newOprnode()と newArgnode()では関数 newExpnode()と同様に malloc(size)で指定した size 分のメモリ要域を xp に代入して newOprnode()では(expnode)xp に、newArgnode()では arg に戻り値を返す。

newArgnode() の引数 argnum は何？

関数 term()では、s を関数 getItem()で取得、termp を Null つまり 0 と宣言し，prefix を 0 と宣言する。symset.c より関数 symsetHas()が true の場合、prefix と s をそれぞ

レポートの書き方について「こういう説明は書いてはダメですよ」と注意しましたが、この部分はそのダメな書き方になっています。延々と中身の説明を書くのではなくて、「この関数は何をしている」と簡潔に述べて下さい。

s.token が sym_lpar と同値な場合に termp を e 関数 expression()で、s を関数 getItem()で取得する。

s.token が sym_rpar と同値ではない、または termp が Null の場合に関数

abortMessageWithToken()を呼び出す。

prefix がないまたは'+'の時は()の内部をそのままにして termp に戻り値を返す。

()の内部が二項演算子の式ではなく、かつ同じ単項演算子が繰り返されている場合に構造体 termp の prefix を 0 に代入し termp に戻り値を返す。

()の内部が二項演算子の式または()の中と外の単項演算子が違う場合に関数 newOprnode に戻り値を返す。

s.token と tok_id が同値の場合で、s.kind が id_func と同値な場合に agp に関数 newArgnode()を代入後、関数 expressionList を呼び出す。その後(expnode)agp に戻り値を返す。

s.token と tok_id が同値の場合で s.kind が id_proc と同値な場合に abortMessageWithToken()を呼び出す。

s.kind が id_func、id_proc と同値でなく、id_undefined と同値な場合に abortMessageWithToken()を呼び出した後、info を宣言した後 termp を関数 newExpnode()で呼び出し、構造体 termp の v.varinf に info を代入する。その後 termp に戻り値を返す。

s.token が tok_num と同値の場合に termp に関数 newExpnode()を代入し構造体

termp の v.intvalue に s.a.value を代入する。Termp に戻り値を返す。

・functions.c

関数 parametar_list()では、s に関数 getItemLocal()を代入する。s.token と sym_rpar が同値の場合に 0 を返す。

このプログラムは function という通り関数に関わっているプログラムだと思われるため、その他のコードを詳しく理解していないと十分に理解することができない。

まあそうなんですけど、各関数にはコメントも少々つけてありますので、グループ内で相談しつつ機能を把握して下さい。