プロジェクト演習
課題3

担当したソースコード

前回から進展が見られないようです。前回の指摘事項に追加、訂正も見られません。どうするのですか？

nextch
printitem
simplestat
statements
strliteral

nextch.c

getTextBuffer関数ではtip構造体のそれぞれの変数を0で初期化している。そんなことはしていません。
freeTextBuffer関数ではtip構造体のメモリ領域を解放している。tip構造体は入力されたデータの先頭1ビットを格納している。
printTextBuffer関数ではisEOFが真か偽によって出力を変えている。isEOFの真偽は次のビットにデータがあるかどうかを判定している。
これでは説明になっていません　次とは？
get_new_line関数ではtip構造体を引数とし、tip構造体の各変数のそれぞれに値を代入している。
nextbyte関数では読み込んだ文字列を1文字ずつ解析している中で次の文字が存在しているかを判別する役割を持つ。nextbyteは文字列を解析する際に使用している。
関数の役割は？
undoch関数ではchとtipを引数とし、tipのungetchが0のときungetchにchを代入している。
このプログラムは全体で、読み込んだ文字列を1文字ずつ解析する処理の次のビットの処理への移行の制御を行っている。
意味不明

nextch.h

nextch.cで用いる構造体、関数名を定義している。

printitem.c

これ以降、前回のレポートからほぼ変更がありません。

shorten関数ではs,limitを引数とし、配列bufに格納する数値を計算し返り値として出力する。
itemToString関数ではbuffer,pを引数とし、tの数値によってテキストを表示する。
printItem関数ではpを引数にとし、pの状態によってデバッグ処理を行っていると思われる。

simplestat.c

chechAssignment関数ではkindの状態によって操作を中断させる。
assignStatement関数ではahead,terminatorを引数としている。chechAssignment関数を呼び出した後、getItemを代入したsの状態によって処理を行い、statmpを返り値としている。
returnStatement関数ではterminatorを引数にしている。構造体rtnpの3つの要素に数値を代入している。statmpを返り値としている。
inputStatement関数では構造体stpの要素に数値を代入している。stpを返り値としている。

forStatement関数では構造体stp,fopのそれぞれの要素に数値を代入している。stpを返り値としている。

statements.c

「構造体」が付いただけ

statInitialize関数では構造体symset_tの各要素の初期化を行っている。
ifStatement関数ではduskulでif構文を解析する場合の処理の制御を行っている。構造体stpを返り値としている。
whileStatement関数ではduskulでwhile構文を解析する場合の処理の制御を行っている。構造体stpを返り値としている。
callStatement関数ではduskulでcall構文を解析する場合の処理の制御を行っている。構造体stpを返り値としている。
printStatement関数ではduskulでprint構文を解析する場合の処理の制御を行っている。構造体stpを返り値としている。
fetchStatement関数ではaheadを引数にしている。aheadの状態によって次に処理を移す関数を制御している。
varsDeclareList関数ではduskulでvarを用いて変数を宣言する際の制御を行っている。varsを返り値としている。
codeblock関数ではtermset,rtnflagを引数としている。termset状態、rtnflagの真偽によって適切な処理を行っている。rootを返り値としている。

statements.h

statement.cで用いる関数名、ポインタを定義している。

strliteral.c

関数addLiteralStringでは入力された文字列を、そのサイズに適したメモリを用意して保存している。
getsymbol.c内の関数get_stringで読み込んだ文字列を受け取っている。

strliteral.h

strliteral.cで用いる関数名、ポインタを定義している。

グループの課題の機能拡張について

if文while文などの文を解析するための処理を行っているため、グループの課題であるrepeat文の実装に当たってstatement.cに変更が必要だと思われる。simplestat.cもしくはstatements.c内にrepeat文を解析する関数を実装し、statement.c内のfetchStatement関数でその関数に処理を移すように変更を加えることが適切だと考える。 **これはその通り。データ構造は必要ありませんか。**
他のグループの課題の機能拡張については自分の担当するソースファイルでは変更する必要がないと考える。

配列の要素への代入を考えると、代入文をなんとかしないといけません。

ソースファイルごとの設問について

simplestat.c

ungetItemは入力を受けていないことを記録しておく変数で、inputStatement()とforStatement()には必要ないため使われていない。 **違います**

statements.c

nodpに一度預けたデータを受け取っている。**？**

代入文とreturn文は、文の終わりが式であるため、その文の次に現れてもよいトークンの集合を引数で渡されています。これをチェックするために、いったんトークンを取り出し、チェックしてからungetItem()で「元へ戻す」という処理を行っています。input文は ')' があれば終わりなので ungetItem()は必要ありません。forStatement() は codeblock() を呼んでいますが、codeblock()はその中で ungetItem() を使っています。forStatement() は、codeblock() の最後のトークンとして 'end' を指定していますので、codeblock()から戻ってから getItem() をわざわざ 1 回呼んでいます。

関数codeblock()は、文列に含まれる文を次々に解析し、それを表す構造体 stnode を得て、数珠つなぎの構造（線形リスト）にします。その際、最も新しく得られた構造体を参照するためのポインタがどこにあるのかを表すのが変数 statmp です。一番最初は、ローカル変数 root から参照する必要があるので、初期値は &root です。新しい構造体を確保したら、そのアドレスを statmp が指す位置に代入します。同時に statmp は、その構造体のメンバ next を指すようにします。
関数codeblock()は最終的に root の値を返しますので、結果、線形リストが一つ返されることになります。