# プロジェクト演習

学生番号：744221
氏名：　榎本 啓希

課題　repeat 文の追加

- 自分が担当したソースコード
  - getitem.c, getitem.h
  - getsymbol.c, getsymbol.h
  - identifires.c, identifires.h
  - idtable.c, idtable.h
  - main.c

**repeat文の実装はまだ完了していないようですが、作業内容もよく分かりましたし、作業を進める上での考察も的確で、大変良いと思います。**
**この調子で進めれば実装は大丈夫でしょう。**
**配列の実装には、グローバル／ローカル変数の情報をどのように管理しているかを把握する必要があります。**

- 演習時間に行った作業

  担当したソースコードに予約語の追加、構文の実行に関係する処理があったかどうかをメンバーに確認し、心当たりのあるものにその該当箇所を大まかに説明してもらった。その後、該当箇所を持っている人には実装を、そうでない人にはサンプルコード作成にあたってもらった。

  私の担当するソースコードの中では identifiers.c に repeat と unitl の予約語を追加するぐらいであったので、その後サンプルコードの作成を手伝った。

  サンプルコードが完成した後、全体の進捗を確認し、全く手が進んでいないようであったため、わからない場合はとりあえず while 文の while を repeat に変更するだけで、テストできる形に持っていこうとした。　**なるほど、現実的な選択で、良いと思います**

  また、github に repeat 用のブランチを作り、時間中に作ったとりあえずのコードを結合した。

- 実施方針に基づいてその方法でうまくいくはずであること

  repeat 文は while 文とよく似ており、構文図に関して、式と文列が逆になっているぐらいであるので、まずは while 文を参考にするところから考えた。

  Xcode で while 文の実行の動向を追うと、while を識別した後、whileStatement によって、whilenode を作成後、構文解析が行われることがわかった。そのため、それらの関数を repeat 用にも作ればうまくいくはずであると思った。

今回うまくいかなかったのは while をそのまま移行しただけであることが大きい。while では、while という文字列を見つけた後式の評価をするが、repeat では until の後に式の評価を行う。この処理を実装しないことには、うまくいかない。コードを見る限りは statement では文字列を構文木にしているだけで、実行はおそらく execute で行うため、順番に書けばうまくいくと考えられる。

whileStatement で言えば、while を読み取ると、whileStatement に入り、expression で式の評価をし、do が続くかの確認をしている。つまり、自分たちは repeatStatement では repeat を読み取れば blockNestPush でスコープを掘り下げ、codeblock に渡す構造は変えずに、その後 unitl が来た後に expression で式の評価を行えば良いと考えられる。また、現在 until を stat_set に入れているが、そのままでは exit_set がないため、必ず”illegal statement” に引っかかってしまうことがわかった。そのため、until を end_set か rtn_set のどちらかに入れるか、新たに作って termset に渡す必要があることが考えられる。

**適切な考察です**

- 他の2つの拡張機能の実装方法について

配列の実装に関して、今までは statement 周りのソースコードを見れていなかったため、文字列を読んでいる段階で条件分岐すれば良いと思っていたが、var が読まれてから呼び出される var 以下を解析しているコードがあることを知った。varsDeclareList 内で”[”のトークンが渡された時、”]”のトークンが出るまでの数値解析（式の解析）をしておけば、配列の読み取りができると思う。ただ、変数を登録する際にどういう構造にするかはまだ解決法がわかっていない。変数をテーブルに追加している部分はわかっているので、次回では変数はどのような形で格納されているのかを追求したいと思う。

**getItem() の結果が整数定数かどうかチェックするだけでOK**

- まだ理解できていない部分について

今回、whileStatement を読んでいて構文解析をしているというだいたいのことは理解できたが、最後に(void)getItem()をしている部分がよくわからなかった。

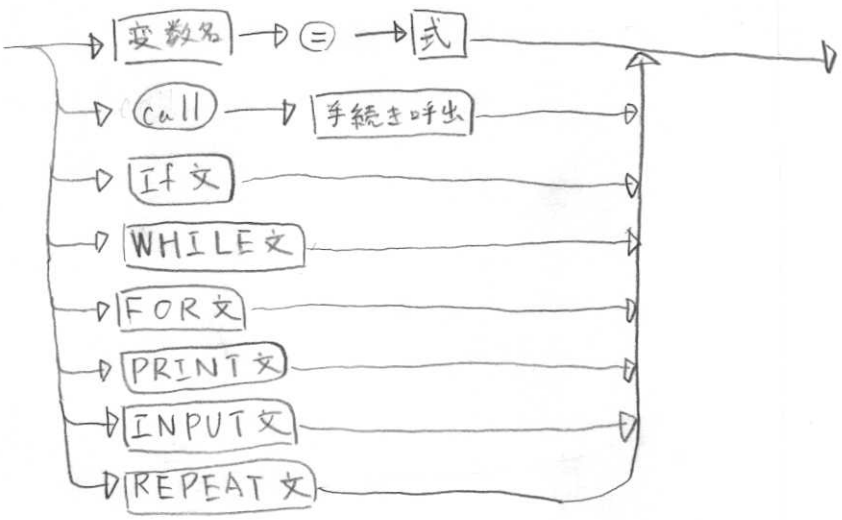
**その数行前で codeblock(end_set, false); という呼び出しをしています。この呼び出しが正常に終わった場合、文列が ”end” を発見して終わっているはずです。codeblock() は最後に ungetItem(s); をしていますので、whileの処理の最後の getItem() の結果は必ず “end” ですから、返り値を無視しても問題ないというわけです。**

# G07の報告書

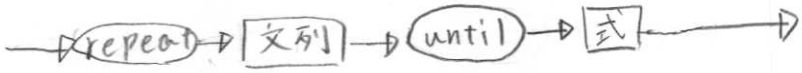
修正した構文図

文

実行文




REPEAT文




**(1)repeat文の実装**
**予約語の追加をする**
**シンボル（トークン）の追加をする**
**repeatのnodeを作成**
**repeatのstatementを追加**
**execute内で実装する**

**実装の方針を説明して欲しいのです。**
**例えば「repeatのnode」とは何？**

**(4)動作しませんでした**

**動作しない、だけでは情報が不足。**
**コンパイルは通ったのか、実行時にエラーが出るのか。**
**実行はできるが想定と違う動作なのか。**

G7