修士論文

C言語穴埋め問題を用いた学習システムの実現

指導教官　　横田 一正 教授

岡山県立大学大学院 情報系工学研究科
電子情報通信工学専攻

有安 浩平

平成22年2月8日

C 言語穴埋め問題を用いた学習システムの実現

目次

1 序論 **1**

1.1 本研究の背景と目的 1

1.2 本論文の構成 2

2 穴埋め問題を用いた学習システム **3**

2.1 穴埋め問題を用いた学習とは 3

2.2 関連研究 3

2.3 関連研究の問題点 4

2.4 システムの要件 4

2.5 アプローチ 6

3 問題の自動生成 **8**

3.1 自動生成機構 8

3.1.1 生成する問題 8

3.1.2 問題生成の流れ 9

3.2 構文解析 9

3.3 問題作成意図 9

3.4 問題作成ルール 12

3.5 問題生成 13

4 出題の自動制御 **15**

4.1 出題手法 15

4.1.1 関係性を用いた出題 15

4.1.2 形成的評価を用いた出題 15

4.1.3 総括的評価を用いた出題 16

4.2 出題の流れ 16

5 自動採点 **17**

5.1 採点上の問題点 17

5.2 採点手法 17

5.2.1 対応テーブルによる変数の比較 18

5.2.2 正規フォーマットへの統一 18

5.2.3 構文木の比較 18

6 実装 **19**

6.1 問題の自動生成 19

6.2 出題の自動制御 19

6.3 自動採点 19

6.4 システムの全体像 20

7 実証実験 **21**

7.1 実験方法 21

7.2 実験結果 21

7.3 考察 21

8 結論と今後の課題 **22**

謝辞 **22**

参考文献 **23**

1　序論

1.1　本研究の背景と目的

大学の情報系学科や情報系企業の研修などでは，プログラミング言語の教育のためにC言語の教育を重視していて，C言語の科目を必修としているところが多い．C言語の学習の初期段階では，概念の理解はもちろん重要であるが，実際にプログラムを作成する過程を繰り返すことも重要である．授業内容を知識として定着させるには何度も学習を繰り返す必要があり，プログラムを作成する過程を繰り返すことでプログラムの動作を理解したりプログラムの作成手順に慣れさせている．

しかし，学習者にやみくもにプログラムを作成させても学習の方向性をを持たないままプログラムを作成することになるので知識の定着や概念の理解のサポートができるとは限らない．そのような問題を解決するために，学習者に問題を出題し解かせる演習システムが多く開発されている．[1][2] これらのシステムでは穴埋め問題を出題し，その穴の部分を考えさせることにより学習を行っている．そうすることにより，ある程度内容が決定されたソースを読むことで学習を効果的に進めることができる．だが，これらのシステムでは問題の作成，出題内容の決定など問題作成者にかなりの労力を必要とする．

また，こうした穴埋め問題生成のための労力の軽減を行うための試験システムも研究されている．[3] ソースコードをシステムに登録することにより，ソースコードの一部をランダムに穴に置換し問題として提示するシステムである．このシステムでは穴にする場所をランダムで決定しているため，学習効果がある出題ができるとは限らないといった問題点がある．

このような問題点を踏まえ，本研究では，問題生成にかかる労力の軽減と学習者に学習を効果的に進めさせることを目標に学習システムの開発を行った．この学習システムでは主に「問題の自動生成」,「自動出題」,「自動採点」の3つの機能を持つ．

問題の自動生成ではソースコードと作成したい問題のイメージをシステムに提示することでソースコードを構文解析し，その構文木の部分木を問題イメージ通りに穴に置換することにより自動的に穴埋め問題を生成する．自動的に問題生成を行うことにより問題生成にかかる問題作成者の労力を軽減させる．

そして，穴埋め問題を総括的評価や形成的評価を組み合わせて効率的な順番

で自動で出題し学習者に解かせる．さらに，構文解析を用いて構文木の比較などを用いて学習者ごとのプログラミングの癖に対応した採点を行う機能の開発を行った．これらの機能により学習者の学習をサポートし，効率的な学習を出来るようにする．

1.2　本論文の構成

本論文では，2 章で関連する穴埋め問題を用いた学習システムについて述べ，それらの問題点と本研究のアプローチを述べる．3 章では問題の自動生成について述べ，4 章で出題の自動制御，5 章で自動採点の詳細について述べる．そして，6 章でそれらの実装について述べ，7 章でシステムを用いた実証実験について述べる．最後に，8 章で結論と今後の課題でまとめる．

2　穴埋め問題を用いた学習システム

2.1　穴埋め問題を用いた学習とは

C 言語などのプログラミングの学習において，アルゴリズムの組み立て方を学ぶこと，プログラミング言語での表現能力を育成することなどが最終的な目的となる．

授業などでは単元ごとに例題や演習問題を与え，例題のプログラムを読ませたり，プログラムを実際に組ませることによりプログラムの書き方や動き方を学習させており，それを繰り返すことで多くのプログラムに触れることができ，学習者に組み立て方や表現力を学ばせている．

このような学習ではプログラムの流れを考えさせながら，プログラムを読ませることが重要になる．プログラムを読むのには構文の知識

また，似た内容の問題を何度も繰り返し行うことで知識の定着を図る目的がある．

2.2　関連研究

穴埋め問題を用いた学習システムとしては香川大の記入式 Web 試験システム DrilLs-L[1] がある．DrilLs-L では問題データベースに登録されたグループ分けされた大量の問題を簡単な出題指定を行うだけでデータベースの中から条件を満たす問題を探し出し，ランダムに出題させる．そうすることで，関連のある問題を何度も繰り返し解かせることにより知識の定着を図っている．DrilLs-L ではウェブブラウザを用いて複数人の同時アクセスにも対応し，予習や復習に使うシステムとして使われている．

また，明治大学の MAX/C[2] も繰り返し穴埋め問題を解かせることで学習を行うシステムである．MAX/C ではアルゴリズムを用いて 1 つの問題から複数のパターンの問題を出題することが可能である．代入操作を学習させる問題などを扱い，代入する値をアルゴリズムを用いて動的に変更している．MAX/C での出題例を図 1 に示す.

宇部高専の JavaDrill[3] ではソースコードを登録するだけで自動的に穴埋め問題を出題し，採点を行うシステムの開発を行っている．このシステムでは JAVA の学習に対応している．このシステムでは穴の位置は乱数を用いて、プログラム中の単語をランダムに穴をあけている．そのため，問題作成の手間はかなり

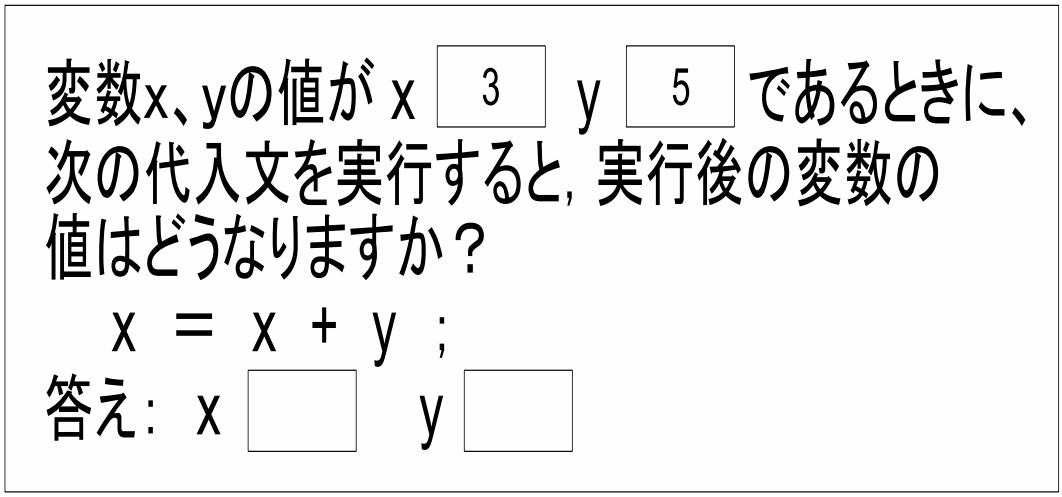


図 1: MAX/C の出題例

軽減されている．

2.3 関連研究の問題点

しかし，2.2 節で述べた関連システムにも問題点がある．

DrilLs-L や MAX/C などのシステムではデータベースに登録やアルゴリズムの記述などを行って問題を作成・登録しているので，新規に新しく問題を作成するときに問題作成者に多くの負担がかかる．また，ある程度システムで扱う問題の記述方法について知識を持っている人でしか問題を作成することができない．そのせいで，記述法を理解している問題作成者に多大な労力がかかる．

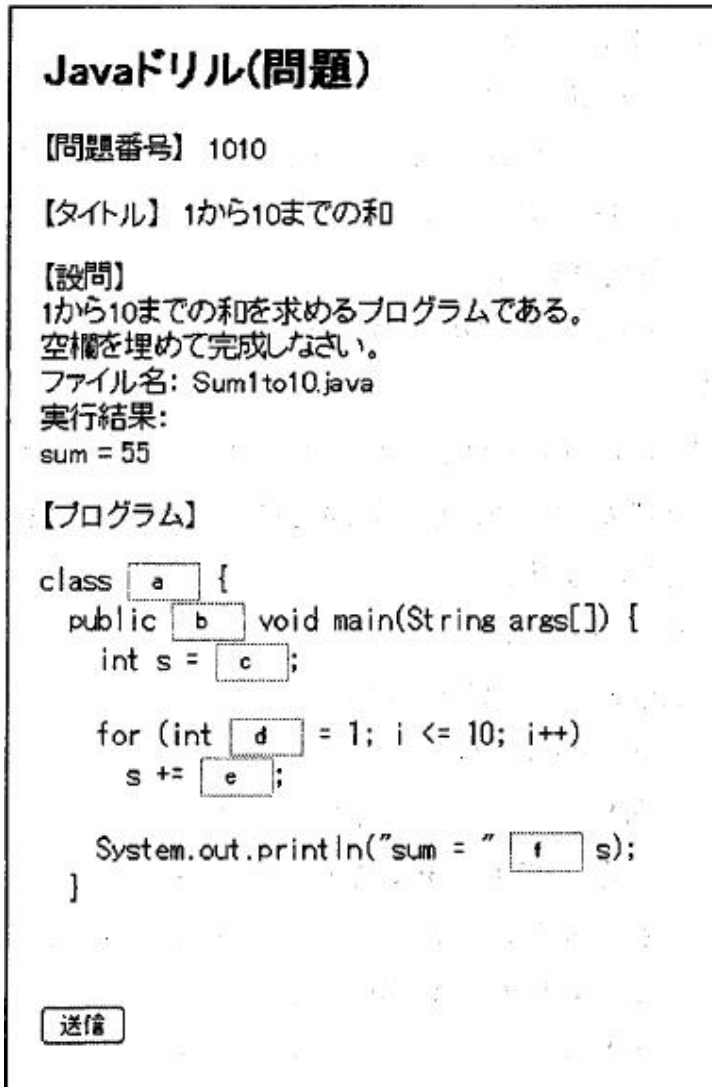
JavaDrill ではソースコードを登録するだけで穴埋め問題を作成することができるので，問題作成者への負担はかなり軽減されている．だが，穴の位置をランダムで決定しているため出題する問題のレベルを制御しづらい．また，同一のソースコードでも穴の開ける位置が変わるだけで，入出力の問題になったり，変数の問題に変化したりレベルと同時に出題範囲も変わってしまうため，意図した学習効果がある問題が出題されるとは限らないという問題もある．

2.4 システムの要件

2.3 節で述べた問題点を基に，システムの要件を述べる．

要件 1 問題作成者の負担を軽減する

学習の進行状況に合わせて多数の演習問題を作る必要がある．しかし，問



Javaドリル(問題)

【問題番号】 1010

【タイトル】 1から10までの和

【設問】
1から10までの和を求めるプログラムである。
空欄を埋めて完成しなさい。
ファイル名: Sum1to10.java
実行結果:
sum = 55

【プログラム】

```
class [ a ] {
  public [ b ] void main(String args[]) {
    int s = [ c ];

    for (int [ d ] = 1; i <= 10; i++)
      s += [ e ];

    System.out.println("sum = " [ f ] s);
  }
```

送信

図 2: JavaDrill の出題例

題作成の作業は手間がかかり，問題作成者には多くの負担がかかる．そこで，なるべく手間をかけないで自動で穴埋め問題を作れる必要がある．

要件 2　問題作成者の意図を考慮した問題の生成

学習目標に沿って穴埋め問題を作成する場合，学習目標を満たすために必要な項目を問題作成意図として問題作成者は持っている．穴埋め問題を作成する場合，その問題作成意図に合わせた内容の問題生成が必要になってくる．そこで，穴埋め問題には問題作成者が定めた問題作成意図に対応している必要がある．また，問題作成意図はいろいろな抽象度の意図があるため，それに対応する必要もある．

要件 3　間違いを考慮した出題

演習では，繰り返し問題を解くことで学習効果を高めることができる．プログラミングの問題になさまざまな種類があり，学習者によってはある特定分野の問題が理解できていないといったことが起こる．そこで，理解できていない分野を推定し，理解できるように導く出題法が必要である．

要件 4　迅速で学習者の差異を吸収する採点

プログラムは繰り返し構文の使い方など，同じ動作を行うプログラムでも書き方が複数あるものが存在する．そういった，複数考えられる記法についてどれを答えても正解になるようにならないと正解なのに間違いと判定されてしまうことが起こる．そこで，その複数の記法を同一とみなし採点を行う必要がある．

2.5　アプローチ

2.4 節で述べた要件を踏まえ，本研究でのアプローチを以下に述べる．要件 1 で述べた問題作成者の負担を軽減と要件 2 で述べた問題作成とを考慮した問題の生成についてのアプローチを述べる．本研究では問題作成者が C 言語のプログラムソースをシステムに入力し，出題したい問題イメージを問題作成意図という形で提示することにより，自動的に穴埋め問題を生成する．

要件 3 で述べた間違いを考慮した出題について，形成的評価と総括的評価を組み合わせて問題 1 問ごとに評価を行い次に出題する問題の制御を行う．また，問題をグループ分けすることにより間違えた問題から，学習者の苦手分野を推測し苦手分野を重点的に行う制御も取り入れる．

要件 4 で述べた迅速で学習者の差異を吸収する採点については，構文解析を

利用した採点を行う．解答のソースと学習者が答えた回答のコードを両方とも構文木に変換し比較する．そうすることにより，構造の差異を簡単に取得できる．また，構文エラーなどの判定も構文解析時に出来るので迅速な採点が可能である．また，違う記法で同じ動作を行うものに対しては統一のフォーマット（正規フォーマット）に統一することで，その差異を吸収する．

3　問題の自動生成

この章では 2.5 節で述べた 1 つ目のアプローチの穴埋め問題の自動生成について述べる．

3.1　自動生成機構

3.1.1　生成する問題

プログラミングの演習問題ではプログラムを一から記述させる記述問題，プログラムのあるまとまりを空欄にしその空欄部分を記述させる穴埋め問題，穴埋め問題の答えを複数の候補の中から選択する選択問題など複数のバリエーションがある．

しかし，記述問題ではプログラムを一から作らせるため，学習初期の学習者にはハードルが高く学習意欲をそぐ可能性がある．さらに，プログラムを一から作らせるためプログラム設計が上手くできず問題作成者の意図した学習効果が得られない可能性がある．また，選択問題では複数の選択肢の中から答えを選ぶので適当に答えても正解になる可能性があり，ここでも意図した学習効果が得られないことがあると考えられる．そのため，今回はプログラムの一部分を穴にしてその部分のコードを書かせる穴埋め問題が有効であると考える．そこで，本論文では穴埋め問題を用いて学習を行うこととする．生成する問題例を図 3 に示す．

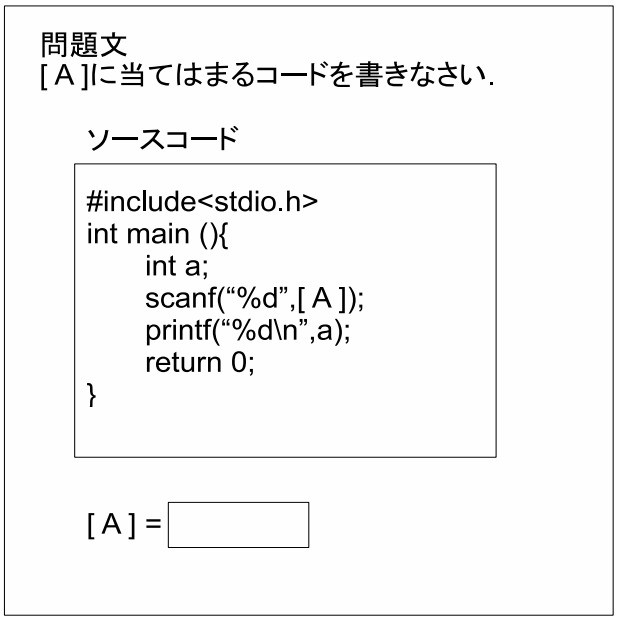


図 3: 生成する問題例

3.1.2 問題生成の流れ

問題作成では図 4 で示す流れで自動的に演習問題の生成を行う．まず，問題作成者は問題にするためのソースコードを用意する．それに対して構文解析をを行い構文木に変換する．問題作成者は後述する問題作成意図を選択する．システムでは問題作成意図に対応した問題作成ルールに変換し，作成した構文木に問題作成ルールを適応することにより，自動的に問題を作成する．

図 4: 問題生成の流れ

3.2 構文解析

C 言語のソースコードを読み込み，構文解析を行い構文木を得る．構文解析した結果以下のような枝を持つ構文木を得る．

3.3 問題作成意図

問題作成意図とは，問題作成者が「こんな目的で問題を作りたい」と考えている内容であり，ある学習目標に対し,「正解できれば，これが理解できている」という指針になるものである．この問題作成意図は，満たしたい学習目標ごとに様々なものがある．

例えば，構文について学習させたい時，問題作成意図としては「出力文が分かっているのか確認したい」,「for 文の引数が分かっているのか確認したい」,「処理の流れが分かっているのか確認したい」等さまざまな問題作成意図があげられる．

今回はプログラミングの学習初期の内容についての考えられる学習意図につ

節の名前	部分木の内容
DECLARATION	宣言
TYPEDEF	型宣言
TYPE	型名
FUNC	関数
GLOBAL	グローバル領域
BLOCK	{} で囲まれた部分
STATEMENT	文
EXPRESSION	式
FORMAT	フォーマット指定子
ARGUMENTS	引数群
ASSIGNMENT	代入
LASSIGNMENT	代入の左辺
RASSIGNMENT	代入の右辺
RVALUE	右辺値
FOR,DO,WHILE,IF,SWITCH	各構文
ARRAY	配列
STRUCT	構造体
MEMBER	構造体のメンバ
NONE	式の省略

表 1: 構文解析木の節

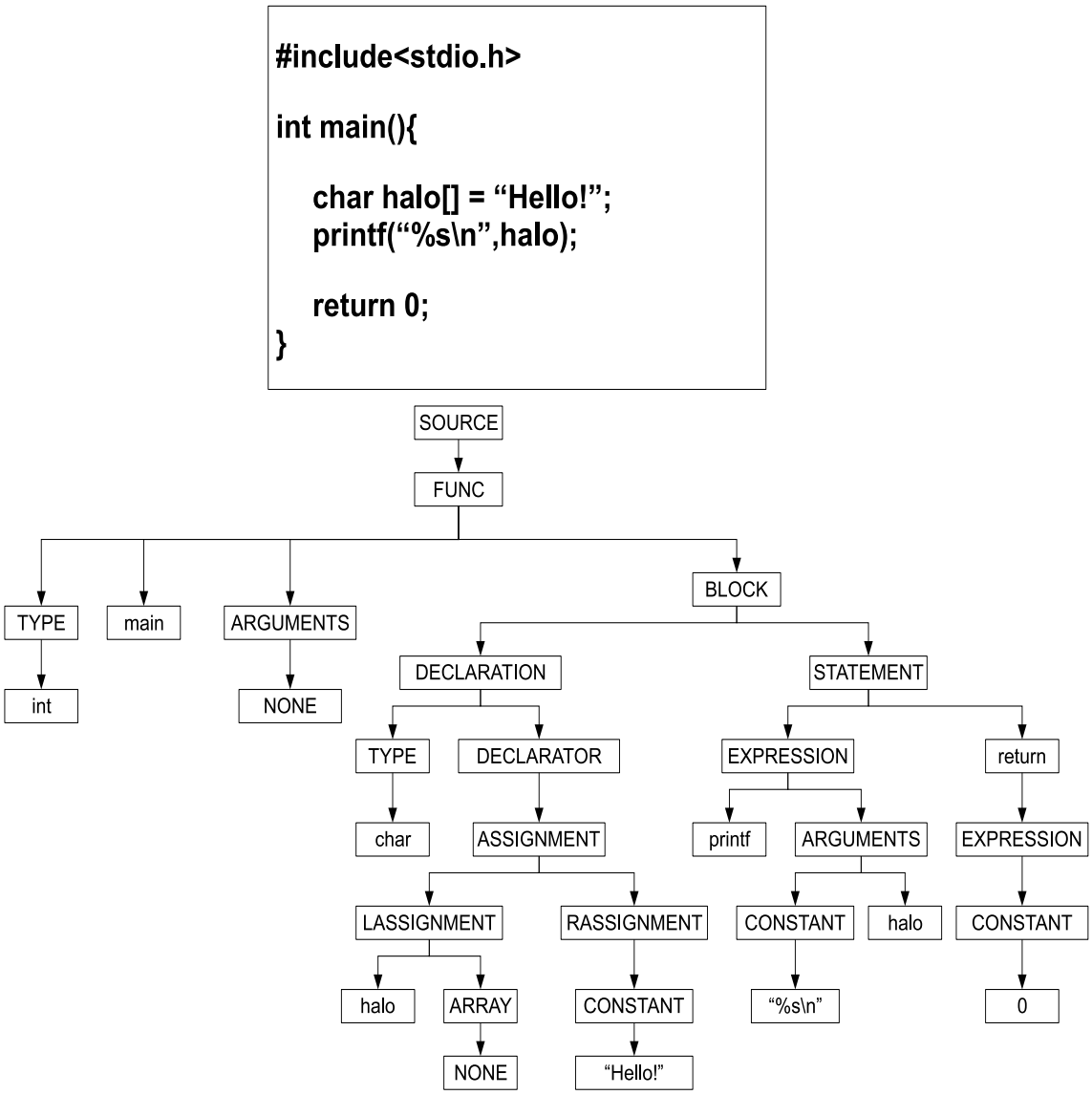


図 5: プログラムの解析例

いて述べる．表 2 は問題作成意図の例である．

表 2 で表された意図は抽象的で、表現が多く記述のバリエーションが多くなる傾向がある。そこで、この問題作成意図は、単純化・具体化するために、以降に説明する問題作成ルールを組み合わせることによって表現する。

入力関数の引数が分かっているか
制御構文の引数が分かっているか
出力関数の引数が分かっているか
関数を使うのに必要なヘッダが分かっているか
必要となる引数が分かっているのか
所定の演算子が使えるか
処理の流れを理解しているか
配列の定義のしかたが分かっているのか
フォーマット指定子が分かっているか
演算子の使い方が分かっているか
どこでどの変数を持ちいらなければならないか分かっているか
返り値が分かっているか
必要な関数が分かっているか
処理内容が分かっているか
自分でプログラムを作れるか
出力結果のイメージができるか
出力関数の動きが分かっているか

表 2: 問題作成意図の例

3.4 問題作成ルール

問題作成ルールは構文木のどの部分を穴にするかを指定する記述である．このルールに従って問題生成部で構文木の部分木が穴に置換され問題が作成される．問題作成ルールは以下のような記述を持つものになる．この記述を複数個組み合わせることにより問題作成意図を表現する。

createblank(枝名, 指定 (, 指定内容))

枝名には表 1 で示した構文木の節が来る．この指定した節以下を穴埋めの穴と置き換える．

指定には”all”または”select”が入り，”all”の場合は枝名で指定した枝の下はすべて穴になる．また，”select”を入れた場合は，後の指定内容で指定したとおりの穴のあき方になる．

指定内容では，枝名で指定した部分の中でも一部分にだけ穴を開けたい場合に

指定内容	実行内容
rootonly	枝名となっている部分のみを書き換える
nameonly	子要素の変数名のみ書き換える
numberonly	変数の値のみ書き換える
conditiononly	条件文のみ書き換える
etc	etc

表 3: 指定内容一覧

指定するのに用いる．しかし，この部分指定では構文解析で分割された単位以下の指定はできない．ここで用いれるものは，rootonly(枝名となっている部分のみを書き換える)，nameonly(子要素の変数名のみ書き換える)，numberonly(変数の値のみ書き換える)，conditiononly(条件式のみ書き換える) などがある。

3.5 問題生成

自動生成には，問題の基となるソースコードと問題作成意図を記述したファイルが必要になる．これら 2 つのファイルをもとに，ソースコードの構文解析を行い，その解析結果に対し問題作成意図を適応し，問題の自動生成を行う．作成する問題は，穴埋め問題とする．穴埋め問題を使う理由としては，穴埋め問題の穴を広くとれば自由記述の問題として出題できるからである．また，選択式の問題のように解答例がないため，勘で答えることが難しいので学習効果が高いためである．

問題は 3.2 節で述べた構文解析の結果と 3.3 節で述べた問題作成意図から作られる．まず，構文解析した結果の変数となる葉に id を付け，変数名から変数名と id の対応表を作る．この対応表では，同じ変数は同一の id で定義される．

そこに選ばれた問題作成ルールを順に適用し，穴をあけていく．穴の開け方は，まず問題作成ルールの枝名を読み，構文木から枝名と一致する部分木を見つける．次に指定のところで記述された内容に従ってその部分木を穴に置き換える．そして，出題には ADEL のテスト機構 [5] での問題の流用が出来るように設計したのでテスト機構でも扱える XML の形式に成型する．このとき，生成する問題には問題ごとに問題 id がふられる．そして、適応した問題作成ルー

ルの種類によりグループ分けされ, 問題データベースに格納される.

構文木に問題作成ルールを適用することで問題を自動生成する。

4　出題の自動制御

4.1　出題手法

本章では 3 章で述べた自動生成により生成した問題の出題手法について述べる．出題は学習効果を高めるために問題の関係性に着目した「関係性を用いた出題」，教育的な評価手法の形成的評価と総括的評価を用いた「形成的評価を用いた出題」,「総括的評価を用いた出題」の 3 つの出題法を組み合わせた出題を行う．

4.1.1　関係性を用いた出題

工学分野の教材は知識を蓄積していくことで学習が進行していく．C 言語でも同様に文法やアルゴリズムなどの知識を蓄積していくことで作ることができるプログラムが増え，様々な要求に対応できるプログラムを書くことができるようになっていく．

このような蓄積型の分野の教材では前に学習した事柄との関連性が強く教材ごとのつながりが強いものが多い．たとえば，for などの繰り返し構文を習う場合，事前に変数についての知識がないと上手く使いこなすことが出来ない．また，同じ学習内容の中でも段階的に知識をつけていくことがあり，順番に解いていくことで理解しやすくなる．ここでの例はポインタでのポインタとポインタ配列などである．ポインタについて先に習うことで，ポインタ配列を学習するときに以前の知識をもとに学習できるので理解が早くなる．

よって，このような関係性をもとに問題を内容ごとにグループ分けを行い，プログラムセットを作成する．ここでのプログラムセットとは，同様な問題内容の穴埋め問題の集合を指す．グループ分けは問題の自動生成時に行われ，プログラムの内容に従って教材作成者が決定する．さらに，同一のプログラムセット内でも穴埋め問題の難易度によりレベルを付加する．このレベルを用いてやさしい問題から順に解くことにより理解がしやすくなる．

4.1.2　形成的評価を用いた出題

形成的評価とは小さなテストなどを繰り返し，学習者の学力の移り変わりを評価する評価手法である．学習の途中に小テストを行いどの程度理解できているのか評価を行う．そうすることで細かく学習者の理解度に対応しながら学習を進めていくことが出来，学習進度と学習者の理解度がマッチしないといった問題を防いでくれる．これを出題に応用することにより，学習者の理解度の変

化に合わせた出題制御が行える．本研究では，この形成的評価を出題に組み込むために，1 問解くごとに 1 問解くごとに正誤判定をし，その場で次に出す問題レベルの切り替えを行うようにした．そうすることにより，学習者の理解度に応じた問題が出題されるようになり，適切な問題が出題される．このレベルの切り替えでは，上（レベルアップ）下（レベルダウン）2 方向の切り替えを行うようにし，最終的に自分の理解度に近い問題が出るように集約されるように設計した．

4.1.3　総括的評価を用いた出題

総括的評価では単元の最後に学習した内容についてまとめてテストを行い，学習者の学力を評価する評価手法である．

最終的な学習目標に対して学習者がどの程度まで理解出来ているのかを計るのに用いることが出来，学習者の学習の全体でみた進捗度合いを把握することができる．これを出題に応用することによりある程度の学習目標にまで到達しなかった学習者に再学習を行わせるなどの制御が行えるようになる．本研究では，

4.2　出題の流れ

本研究ではこれまで述べてきた出題手法を組み合わせた独自の出題手法を用いている．出題の流れは図**??**で示されるような形で出題されていく．

図 6: 出題の流れ

5　自動採点

5.1　採点上の問題点

穴埋め問題の穴を大きくした場合，プログラムによっては複数の解答が考えられるものがある．これは，C 言語では違う記法で同じ動作を書くことができるために，学習者のプログラミングの書き方や知識の違いから起こる．そういった解答に対して ADEL のテスト機構 [5] では，複数の解答を問題作成時に問題作成者に登録してもらうことで対応してきた．さらに，自動生成システム [4] においても複数パターンの解答がある場合には問題作成者に手動で登録してもらうという手法をとっていた．しかし，この手法では解答のパターンが多く考えうる場合，すべてのパターンに対して解答を登録してもらうことは問題作成者に大変多くの負担がかかっていた．また，問題作成者任せで複数パターンの解答に対応していると問題作成者が登録し忘れた場合，回答自体は正解なのんに不正解になるといったことが起こりうる．これでは，問題ごとに採点基準が異なってしまう危険性が考えられる．

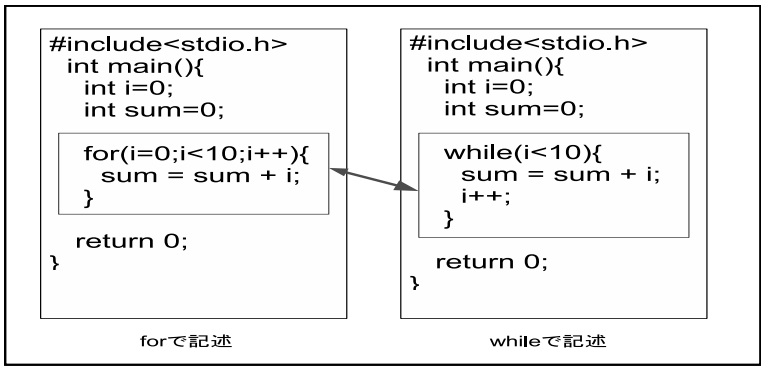


図 7: 違う記法で同じ動作のプログラム例

5.2　採点手法

そこで，5.1 節で述べた問題点について対応した採点手法について述べる．採点機構では,「対応テーブルによる変数の比較」,「正規フォーマットへの変換」,「構文木どうしの比較」といった 3 段階の流れで採点を行う．

学習者が解答してできたソースを構文解析し構文木の作成を行う．そして, 問題作成時に作成した構文木と比較を行う．また，変数の位置をすべて穴をあけた場合は問題リソース作成時につけた id をもとに，記号表により，学習者が入力

した文字が問題作成者の意図した文字列でなくても，必要な場所にすべて同じ文字列が入っていた場合正解とする

5.2.1 対応テーブルによる変数の比較

構文解析を行うときに変数に id をつける．変数の宣言部を解析時にそれぞれの変数に id をふり，変数の対応テーブルを作成する．処理部分に変数が出てきた場合，そのノードの id を対応テーブルに書き込む．

5.2.2 正規フォーマットへの統一

違う記法で似た動作を行うコードに対して，ある 1 つの記法に統一を行う．今回対応したのは「繰り返し構文」,「分配法則」,「結合法則」である．

5.2.3 構文木の比較

2 つの構文木の比較を行う．深さ優先にノードを比較していきルートノードから順に対応するノードを決定していく．対応しないノードが現れた場合，そこを間違いとして学習者に提示する．

6 実装

6.1 問題の自動生成

構文解析はANTLRを用いて作成した．ANTLRを用いた理由は優秀なベンチマークがあり，デバッグ環境も充実していたからである．また，問題作成ルールや問題作成意図の解析機構，問題生成機構はJAVAを用いて実装されている．

テスト記述

program_set　問題のグループを持つ要素で，問題グループを識別するid属性を持つ．子要素には複数のitem要素を含む．

item　問題候補のルート要素であり，固有の識別子であるid属性と問題の形式を表すtype属性をもつ．item1つが問題1問に相当する．子要素にquestion要素，response要素，hints要素，explanation要素，evaluate要素をもつ．

question要素　問題の内容を記述する要素で，穴が開けられたソースコードはここに記述される．

response要素　提示の際に解答の候補として記述され，穴埋め問題の解答欄の数に比例する．

hints要素　問題に対するヒントが記述されている．

evaluate要素　採点に関する事柄を記述する要素．子要素にfunction,correct,score,weight,pointを持つ．

correct要素　正解となる文字列を記述する．この要素と解答を比較し正解判定を行う．

point要素　この問題を正解すると加算される点数を記述する．

explanation要素　採点後に表示される文字列を記述する．

6.2 出題の自動制御

すべてJAVAを用いて実装されている．

6.3 自動採点

すべてJAVAを用いて実装した．

6.4　システムの全体像

自動生成機構で問題作成者が問題を作成し，問題データベースに登録する．問題データベースから学習者に応じて出題機構が問題を選択し出題する．学習者からの回答と問題からプログラムを作成し構文解析にかける．解答のソースを構文解析したものと比較を行い正誤判定を行う．

図 8: システムの流れ

7　実証実験

出題法の有効性を調べるために実証実験を行った.

7.1　実験方法

研究室の学部3年生から修士1年生の17名を対象に実証実験を行った．提案手法のグループとランダムに出題した手法の2グループに分け，その手法で解いてもらい，その後同じ出題順・同じ問題内容の共通問題を解いてもらい，ログを解析して正解率がどのように伸びているのか調査した.

7.2　実験結果

7.3　考察

8　結論と今後の課題

謝辞

本研究を進めるにあたり、懇切な御指導、御鞭撻を頂いた横田一正教授に深く感謝します。

また、本研究に関し多くの有益な御助言、御助力を頂いた國島丈生助教授、ならびに岡山理科大学の劉渤江助教授に感謝致します。

最後に日頃から有意義な御助言、御協力を頂いた知能メディア工学研究室の諸氏に厚く御礼申し上げます。

参考文献

[1] 萩原秀和，富永浩之，松原行宏，山崎敏範，” ドリル練習に基づく Web 型試験システム-情報基礎科目への応用と運用試験-，” 信学技報， vol.102, no.388, ET2002, pp.57-62, 2002.

[2] 玉木久夫，疋田輝雄，井口幸洋，小島崇輝，早川智一，”MAX/C：WEB 上の C プログラミング実習システム，” 全国大学 IT 活用教育方法研究発表会, E-10.

[3] 内田保雄, ” 初級プログラミング学習のための自動作問システム,” 情報処理学会研究報告, Vol.2007, No.123(20071207) pp. 109-113 2007-CE-92-(16).

[4] 池田絵里，”C 言語学習のための穴埋め問題自動生成システム，” 岡山県立大学 卒業論文,2008.

[5] 西輝之, 延原哲也, 劉渤江, 横田一正. ” 適応型 e ラーニングに必要な診断的テスト機構”. DEWS2006.

[6] ANTLR. ANTLR Parser Generator. http://www.antlr.org.