東大理系1年生むけCSスパルタ教育 (プ会2007.5)

久野　靖*

2007.5.22

1　はじめに

□ 東大1年次コンピュータ教育について…というわけではなく。

□ 自分がやってきたこと…頼まれて(非常勤)1年生にプログラミングを教える(自分の趣味に合ってる)(1年後期)(選択科目)

□ 2005年度まで→「好きにやってください」で好きにやっていた(学生はどの先生のクラスかを選択できる)

- どのクラスを取るかは「闇魔帳(?)」を見て学生が決める→甘い先生のクラスに殺到→定員を設けくじ引き

□ 2006年度→理系対象「クラス指定科目」化(その時間帯はその科目しか入れない…必修ではないが取らなくても他の科目は取れない)

□ クラス指定なので学生は先生を選択できない→「教育内容を標準化します」「『情報科学』という科目名としてCSを教えます」

- 「標準化する内容」の表現手段→「標準スライド」(パワーポイント絶対主義)(私はパワポで授業する気はないんですけど)
- 「使用言語はRubyに統一します、ただし'06のみJavaを許容」
- 夏休みになってもスライドができてこない…内容が決まらないと準備できないんですけど?　(結局、9月になってから小出し)
- 開けてびっくり!　ものすごいスパルタ教育な内容!
- でもやってみたら面白かった→ここに報告させて頂きます

2　標準スライド

□ 東大の関係する先生がたが分担して作成

- 各先生のキャラが色濃く出ている(久野に竹内先生や玉井先生に成り替われと言われても…)
- 従って、連続性はほとんどない。たまに、同じ内容が複数回出て来てアプローチが違っていることがある…
- 内容は、かなり幅広くカバーされている
- レベルは、かなり高い(ルンゲクッタ法の導出なんて自分では絶対できないんですけど…)

□ #1:　対象のモデル化とデータ構造(1)(竹内) --- 基本データ型、配列、レコード、オブジェクト、メソッド、Rubyを動かす

- 制御構造は全くやらずにデータ型を全部やってしまう…

□ #2:　〃(2)(竹内) --- 連想メモリとハッシュ、可変配列、再帰的定義、リスト、木、再帰的データ構造、スタック、キュー(とその実現/操作)、グラフ

- コードはほとんど書かずにデータ構造を全部やってしまう…
- 久野の考えとしては、どのデータ型、データ構造も実際にコード書いて使い慣れつつ学ばないと無意味と思っている
- こういう風に全部まとめて講義してしまうというのは普通なの?

□ #3:　データと計算(1)(玉井) --- 情報の表現、名前をつける、日本人の名前、他国の名前(フランスでは500の名前から選ぶことになっていた)、生物の学名、内包と外延、単射/全射、文字列の等しさ(Ruby)、符号化、距離、階層構造、ハッシュ(Ruby)、属性、クラス、データベース、表、計算としての関数/写像(Ruby)、論

*筑波大学大学院経営システム科学専攻

理値、分岐、else-if の連鎖、関数の合成、変数、代入、オブジェクト、状態 (の有無)

- ものすごい量の新規概念の連打じゃないですか?

□ # 4: 〃(2)(増原)+オートマトン (萩谷) --- 表で表せない計算 (無限性)、演算の組み合わせで表せない計算 (階乗、コラッツ予想)、再帰計算、再帰データ構造に対する計算、関数の計算過程 (書き換え、状態遷移、末尾再帰、繰り返し、不変量、ベクトル演算、文字列検索、有限オートマトン、言語、認識器としての有限オートマトン、実装 (Ruby)、正則表現、非決定的オートマトン、決定化、最小化

- 2 回ぶんの内容だと思うのですけど…
- やはり、プログラムを動かしながら対応する概念を学ぶようにして、学ぶ概念の数を絞らないと頭に載らないと思う…

□ # 5: アルゴリズムと計算量 (増原) --- アルゴリズム (前記にもやっているので復習)、アルゴリズムの定義、役割、複数のアルゴリズム、実行時間の違い、平方根 (数え上げ、二分法、ニュートン法)、フィボナッチ (再帰定義、数え上げ、行列積)、最大公約数 (数え上げ、互助法)、整列 (単純選択法、マージソート、ビンソート)、課題 (実行時間計測)、(時間) 計算量とその求め方、空間計算量

- これくらいだと久野の趣味にも合うような (趣味の問題?)…ただしプログラムはもっと多く掲載して欲しいが

□ # 6: 数値解析 (1)(中尾) --- 代数演算と数値計算、実数データ表現、誤差 (丸め、桁落ち、情報落ち、打ち切り誤差)、数値積分 (台形、シンプソン)、常微分方程式、Euler 法、Taylor 展開、Runge-Kutta(2 次、4 次)、Runge-Kutta-Gill、連立微分方程式、物理シミュレーション

- Taylor 展開から Runge-Kutta 法を導くというのはつらい…

□ # 7: 数値解析 (2)(中尾) --- 連立一次方程式、Gauss 消去法、Gauss-Jordan 法、ピボット選択、反復法/Jacob 法、ノイマン級数と収束、乱数、一様乱数、正規乱数、疑似乱数、モンテカルロ法、円周率、ランダムウォーク

- 疑似乱数のアルゴリズムはやらないんだ…
- 最初は数値解析は「やりすぎ」と思っていたが、実際にやってみるとそれなりに興味を持ってもらえる (東大理系だと数学は得意だから?)

□ # 8-9: パターン認識 (萩谷) --- 「等しい」と「似ている」、動的計画法、アラインメント、トレースバック、隠れマルコフモデル、いかさまカジノ、評価問題、前向きアルゴリズム、復号問題、推定問題、Viterbi アルゴリズム

- 遺伝子プログラミングあたりがネタらしい、萩谷ワールド
- アラインメントと Viterbi をプログラミング演習させるという点ではまあまあと思う。ただ、これがパターン認識なのかどうか?

□ # 10-11: 言語処理系と仮想機械 (山口) --- 式木、文字列→字句解析→トークン列→構文解析、有限オートマトンによる字句解析、BNF と構文木へのあてはめ、タイプ 0〜3 文法、再帰下降解析 (Ruby)、式木の表示、コード生成、ED21(シミュレータ) の命令と動作、式木から機械語の生成、仮想機械、仮想機械のインタプリタ

- 言語全体の木はやらない (式木だけ) というのが不満。ED21 の命令セットも説明が手間取りそうなわりにあまりすごくない感じ

□ # 12: プログラミング言語 (増原) --- 言語以前、言語の歴史、命令型 vs 宣言型、オブジェクト指向、スクリプト、翻訳/解釈実行、VM、具体例 (FORTRAN、C、C++、Java、C のプログラム例、アセンブリ言語、アセンブリ言語の書き換え、ISA の違い

- こういうさまざまな言語の話は久野も好きだしやりたいけど…

3 久野の方針

□ 「内容が CS だろうが (CS だからこそ) プログラムを書いて CS の概念を実際に動かして体験しなければ身につかない」

□ 「プログラミングを身に付けるには実際にコードを書くことに時間を費すことが不可欠」

□ 以上の考えから、前年度までの方式を踏襲

- 各回とも、時間の中でプログラム 1 個は打ち込んで動かしてもらう

- 各回とも、その時間に動かしたプログラムを「出席」として提出してもらう (A 課題)(メールで提出)
- 各回とも、「次回までの課題」としてプログラムを 2 個くらい書いて出してもらう (B 課題)(メールで提出)(冬休み前まで)
- 「次回までの課題」の内容として次回講義でタネ明かしになるものを入れておく
- A 課題 B 課題とも必ずアンケートを入れてフィードバックをもらう

□ 以上の枠組みを前提として、組み立ては「プログラムを書く上で前提知識の少ないものから順に」

□ 言語は Java にさせてもらった (前年度までの課題も流用)

□ 「金 2 限」担当→学生は理 2・3 類 (農・生物・医学) →あまり「プログラミングやろう」という感じではないらしい

□ 資料、学生のレポートなどは以下にある (レポートは公開すると最初に宣告)

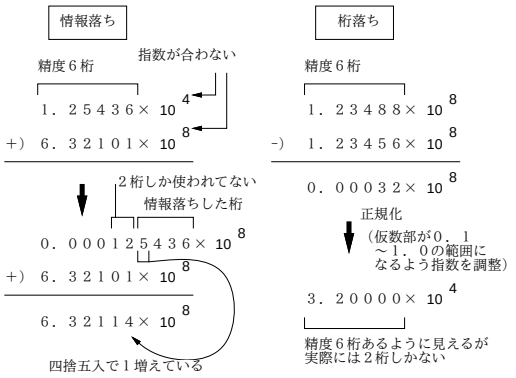
http://lecture.ecc.u-tokyo.ac.jp/~kuno/is06/

□ 以下では進行を追って内容構成と様子を報告

4 プログラミング入門部分

□ # 1: プログラムの基本概念

- アナログとディジタル、コンピュータとディジタル情報、モデル化とコンピュータ、アルゴリズム、変数と代入 (手続き型計算モデル)
- プログラミング言語、Java 言語のプログラム例 (華氏摂氏変換)、打ち込んで動かす (A 課題)、オブジェクトと値 (簡単に)
- 数値の表現、十進と二進、負の補数、浮動小数点、数値表現の限界と各種の誤差、NaN、∞



- B 課題: 限界や誤差の影響を見られるようなプログラム

□ Java コードを打ち込んでコンパイルして動かすまでで一苦労だが、これをやらないと始まらないのでひたすらやってもらう。TA と私で走り回ってエラーなどの面倒を見ることでなんとかする

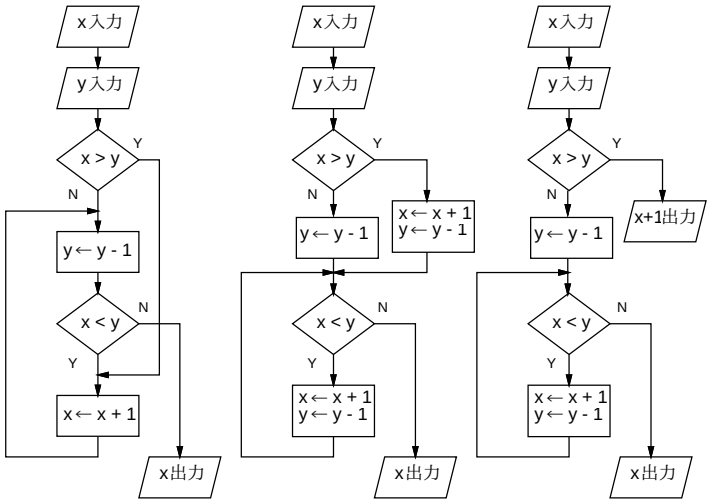
□ A 課題の反応: 訳の分からないものを打ち込まされるのはつらいという意見もあったが、ともかく動かせたことで好意的な反応

□ B 課題の反応: 2 進数の限界や浮動小数点の誤差を試すのは多くの人が考えてくれて成功。きつい、大変という意見はもちろんある

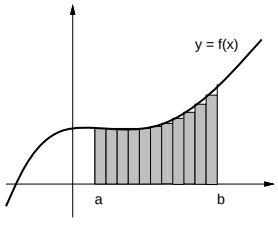
□ 誤差や限界は標準スライドでは数値解析の回に主に出て来るが、初回にできることは「計算」程度なのでそれだけでできる B 課題として選んだことはよかったと思う

□ # 2: 制御構造

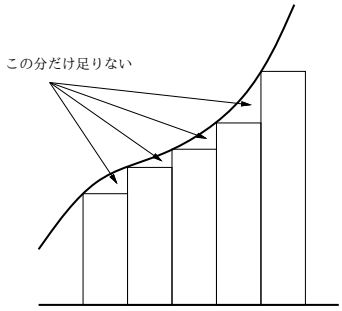
- 以後毎回、前回の主要な B 課題の回答例/解説を掲載
- 基本的な制御構造、連接、分岐、while ループ、任意のフローが基本構造の組み合わせにできる話



- if 文、絶対値の例題、(A 課題)2 数の最大、3 数の最大、正/負/零の表示
- 繰り返しと while 文、数値積分 (柱状公式)、for ループ、台形公式



- 制御構造の組み合わせ (入れ子)
- B 課題: 数値積分の誤差を減らす、GCD の計算、ループで計算するいろいろ (sin/cos の級数とか)、素数判定、素数列挙



□ 初回で数値を単に計算するのはできたはずなので、この回から制御構造を頑張ってもらう。スパルタだが内容が多いので

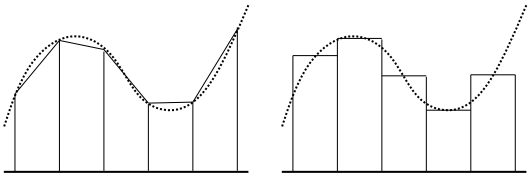
□ A 課題の反応: コンパイルエラーで大変という人と論理エラーで大変という人といる。プログラムの難しさに早くブチ当たってもらうつもりなのでこれでいいと思う

□ B 課題の反応: そろそろしんどいと思う人はレポートが最小限になるが、色々考察してくれる人も多数あり。素数列挙は「10 秒でいくつまで列挙できるか」という内容なので工夫して楽しんでくれている人がそれなりにいる

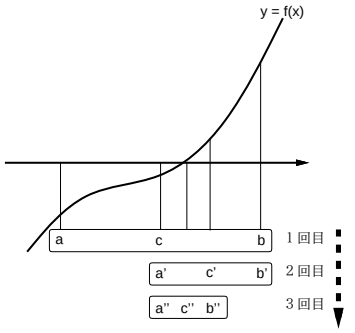
□ ここまでやってくれた人は制御構造は使えるようになったと思われる

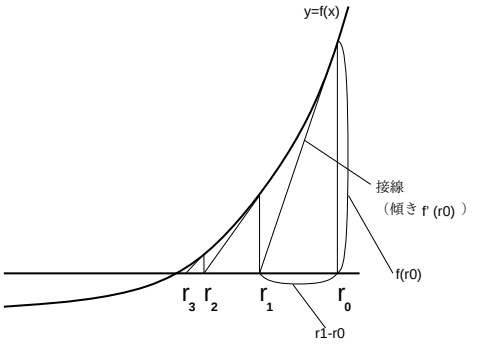
□ # 3: さまざまなデータ型

- 枝分かれの課題解説で else-if 連鎖を説明
- 数値積分の補足としてシンプソンの公式を導く (台形と中点を 1:2 で混ぜるとシンプソンになる)

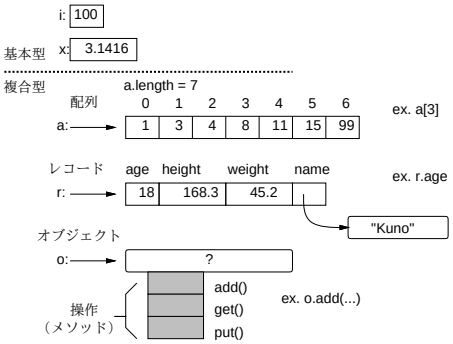


- $f(x) = 0$ の解を求める --- 数え上げ、区間 2 分法、ニュートン (数え上げが例題であと 2 つは A 課題)





- データ型、基本データ型 (数値各種)、構造データ型 (レコード、配列、オブジェクト)



- 素数列挙課題でさまざまな速度向上の工夫について解説
- B 課題: 配列操作 (最大/最小、単純選択法/比較交換法による整列)、素数列挙で既出素数を配列に保持する改良、エラトステネス

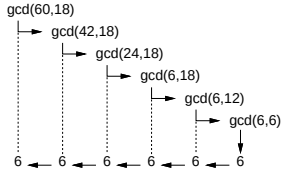
□ A 課題の反応: 今回は「ちょっと手直し」でなく 2 分法かシンプソンのコードを自力で書くので大変だったという意見多数 (まだ身についていない?)

□ B 課題の反応: 相当大変だという意見が多くなってきているが、書けている人のプログラムは相当書けている感じ。素数列挙の改良がやりたい人、整列をやりたい人ともそれなりにいるしどちらもだいたい書けている。つまり、ちゃんとトレーニングすれば 3 週で配列を操作するプログラムは書けるようになるということ (東大理系の場合)

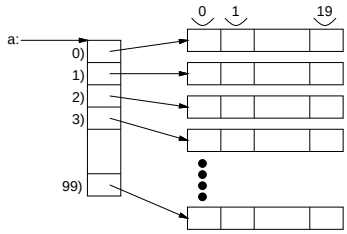
5　中級:抽象化と計算量

□ ＃4:　さまざまなデータ型

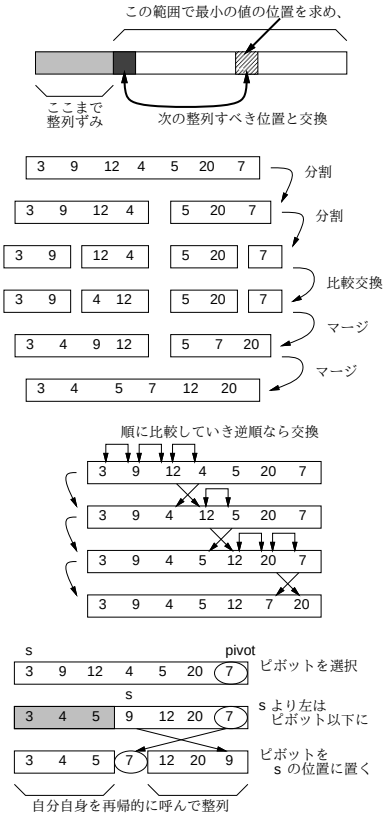
- 手続き (関数) と抽象化、パラメタ、副作用、再帰関数 (例題)



- (A 課題) 再帰関数を書く (階乗、フィボナッチ、組み合わせ、2 進表記)
- レコード型、2 次元配列、レコードの配列、画像、PPM 画像生成 (例題)



- アルゴリズムと計算量、各種整列法 (単純選択法、バブルソート、マージソート、クイックソート、ビンソート、基数ソート)、最大公約数の改良、フィボナッチの改良、組み合わせの改良



- B 課題:　さまざまな図形を画像に追加する (塗る) 手続き、好きな絵、複数アルゴリズムの計算量の計測と比較

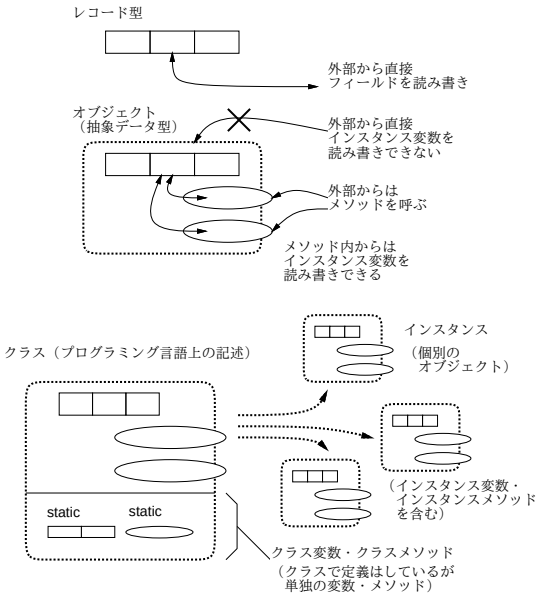
□ A 課題の反応:　関数や再帰は納得したが画像は難しいという意見が多い。画像のデータ構造はやはりこれまでに無い新しいものだからということか。

□ B 課題の反応:　画像はどうやっても独自の内容になるので入れているが、皆それなりに工夫してくれている。計算量の方はわかっていて色々できる人とそうでない人の 2 つに分かれているかも

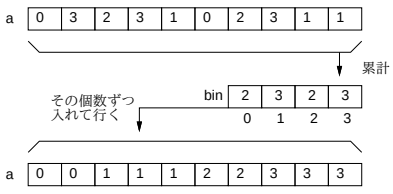
□ アルゴリズムにより計算量が違って来ることを身をもって分かってもらうという目標は達成されているように思う

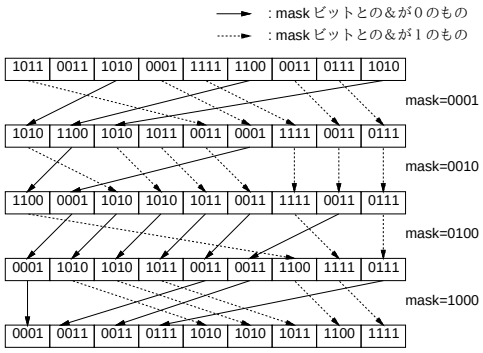
□ ＃5:　オブジェクト指向

- オブジェクト、クラス、インスタンス、API



- API ドキュメントを読む (演習)、文字列の加工 (例題、A 課題)
- コマンド引数、より多様な文字列操作
- クラスの構文、有理数クラスの定義 (例題)
- 計算量つづき --- 各種整列の計算量、最大公約数の計算量、フィボナッチ改良版の計算量





- B 課題： 文字列操作、有理数クラスの機能追加、さまざまなクラスを作る (仕様は与える)、任意課題

□ A 課題の反応： オブジェクトで文字列を扱うというこれまでの数値と違う内容→とまどいがあるが、好意的な人と難しいという人

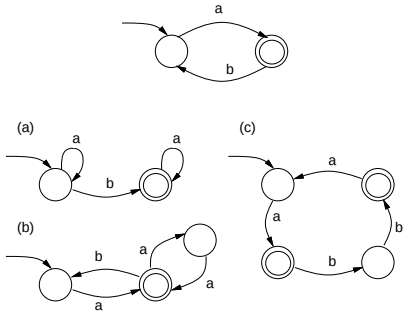
□ B 課題の反応： 文字列操作ものは何とかなっている感じ。復素数クラスを作ってクラスが作れるようになった人もかなり。さらに独自なクラスを作っている人は「離陸した人」という感じ

□ ここまでクリアしてきた人は Java が十分使えるようになっている感じ。この先は「いよいよ」さまざまな CS 的話題に割くということか

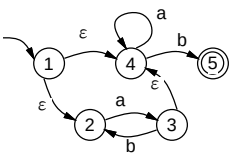
6 CS スパルタ教育

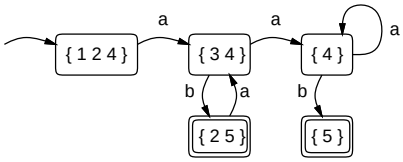
□ # 6： オートマトンと再帰データ構造

- 有限オートマトンとオートマトンに対応するデータ構造+たどる処理をパッケージしたクラス (これを別のオートマトンに変更→ A 課題)

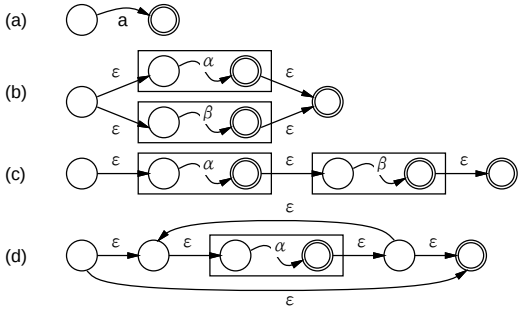


- 非決定性オートマトン、決定化アルゴリズム

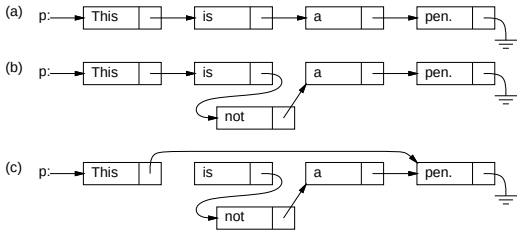


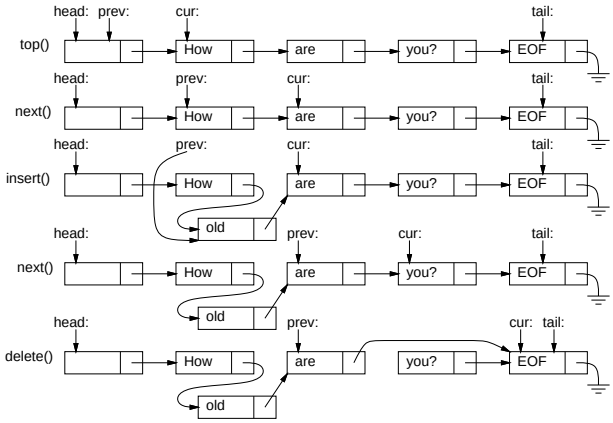


- 正則 (正規) 表現とオートマトン (正規表現→オートマトン)

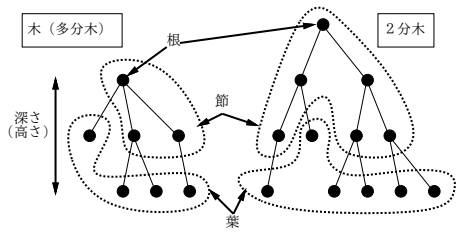


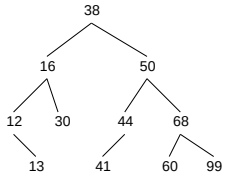
- 動的/再帰的データ構造、単リスト、単リストを用いた行エディタ





- 表と探索、2 分探索木





- B 課題: オートマトンの決定化、行エディタへの機能追加、探索の実現

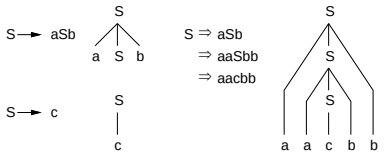
□ A 課題の反応: オートマトン自体は分かったという声が多く、こういうものを考えて頭を使うのは嫌いでなさそう。

□ B 課題の反応: オートマトンで頑張った人、エディタで実用ぽく工夫するのが面白かった人、表探索で時間計った人それぞれあり。

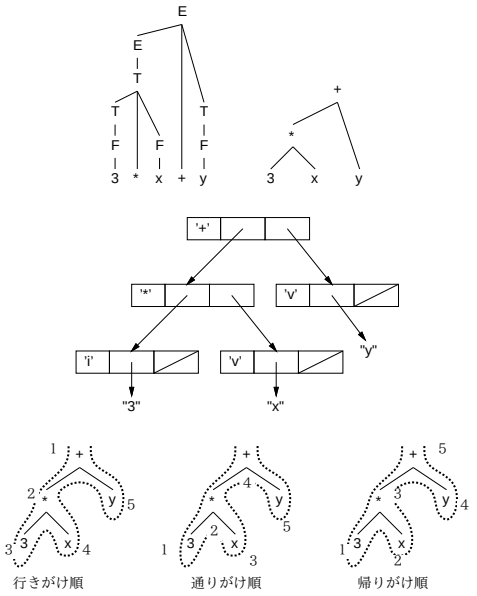
□ かなり「アルゴリズム」というより「CS」になってきた感じで、これが面白いと思ってもらえているなら成功という気分

□ # 7: 文法、構文木、スタックとキュー

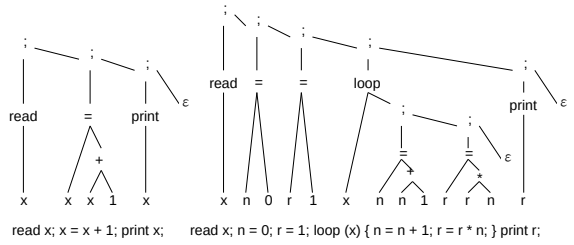
- 生成文法 (タイプ 0〜3)、構文木と構文解析、式の文法



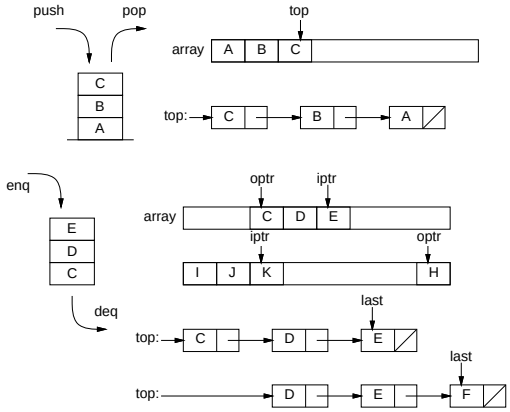
- 抽象構文木と式木、式木の打ち出し (例題→ A 課題)、前置/中置/後置記法とたどり順



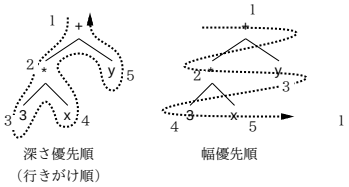
- 「簡単な言語」の抽象構文木

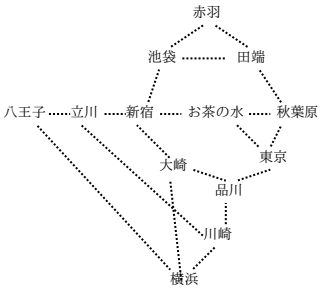


- スタックとキューの概念と実装例 (配列版)、スタックとキューによる構造のたどり



- B 課題: 式木の出力を前置/後置記法にする、スタック/キューの連結リスト実装、鉄道路線図のグラフをスタック/キューでたどる、スタックを 2 つ用いた行エディタの実装。自由課題もあり

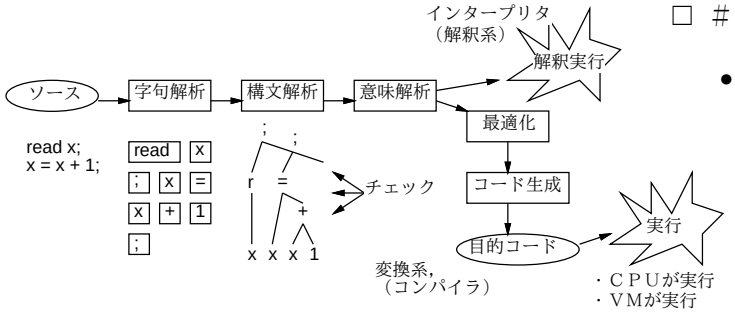




□ A 課題の反応: 構文木は分かったという感想で好意的→オートマトンもそうだが、視覚的に分かるものは好評 (再帰手続きでたどってるのだが、そのことには何ら抵抗ないらしい)

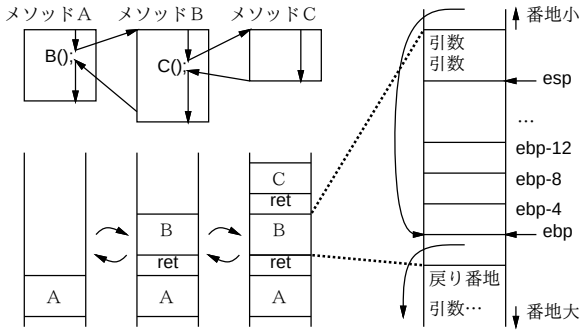
□ B 課題の反応: あんまり頑張りたくない人はスタック/キューのリスト版実現や木の打ち出し。エディタとか路線図の人も。一番やりたい人は自由課題でいろいろ (電卓、ハノイの塔、など)

□ 自由課題でがんばってる人になると、情報科学科の 2 年途中に編入でもいいんじゃないかとか思えてしまう。でもまあ、他の分野に進む人がこれだけ CS もできるというのはすばらしいということで…



□ # 8: 言語処理系の全体像

- 言語処理系の各フェーズ (字句解析、構文解析、意味解析、解釈実行/コード生成、機械語、VM、アセンブリ言語
- 抽象構文木の解釈実行 (A 課題)
- Java の例外機構 (入れる場所がなかったので)、
- x86 アセンブリ言語出力、仮想マシンアセンブリ言語出力



- 仮想マシン用アセンブラ、仮想マシンの実装
- B 課題: 木のインタプリタで命令を増やして実行、仮想マシンの性能計測や改良

□ A 課題の反応: 木構造が動くのは面白いという意見と、もうこんなに難しいのはいいやという意見 (やりすぎたか…)

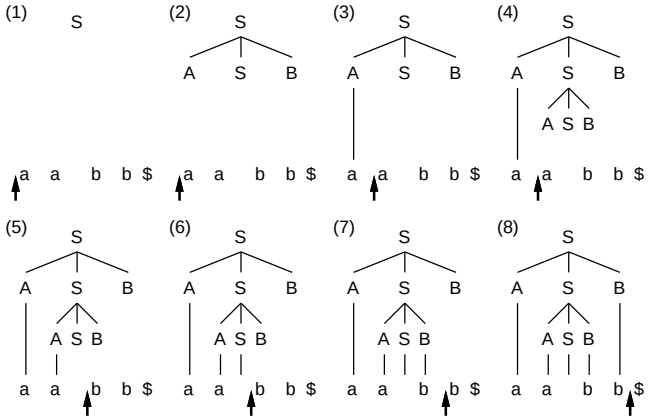
□ B 課題の反応: 多くの人が限界という感じでとりあえず木を動かす課題を選択 (やりすぎたか…)。好きな人はアセンブラだろうが VM だろうがどんどん勝手にやってくれている…

□ 結局、自分の趣味なところに来ると全部教えたくなってしまいやりすぎるといういつものパターンに陥っている気が…

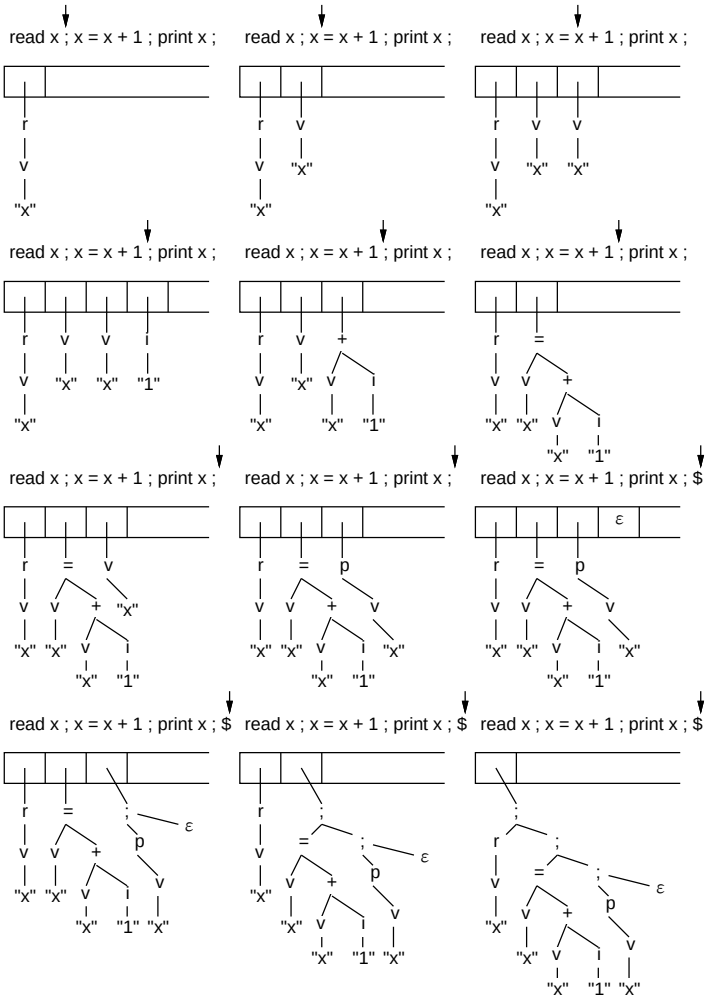
□ この回が年内最終で、この回からは A 課題のみとなる (あとは冬休み課題を紙のレポートで出してもらう)

□ # 9: 構文解析と字句解析

- 文脈自由文法の First、Follow を求める、下向き解析、再帰下降解析、簡単な文法の解析 (課題 A)



- LL(1) 文法、文法の書き換え
- 簡単な字句解析器 (手作り)、簡単な言語の認識器
- 意味スタックと抽象構文木の組み立て



□ A 課題の反応: むずかしい… (一部の人を除く)

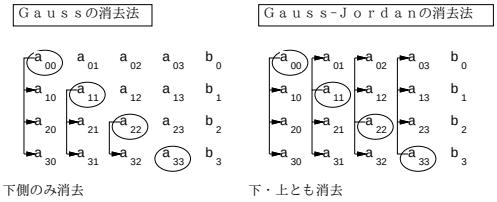
□ 再帰下降解析の原理を納得するのはそーと一大変なのだろうか (やっぱり)。自分としてはコンパイラ全部作れるように説明するには構文解析は落とせないので一番楽な再帰下降解析と思ったのだけど、スパルタすぎましたかね? (代替案としてはテーブルドリブン LL(1) 解析器を用意するとか?)

7 落ち穂拾い

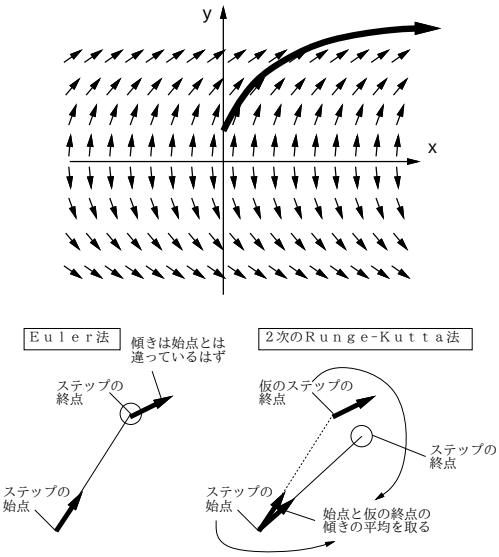
□ 新年になってから 3 回→標準スライドにあってまだカバーしていないものをやり、時間が余れば Java ならではなプログラムも作らせたい (まだまだ欲張っている)(A 課題はどれでもいいことに)

□ # 10: 数値解析の続編

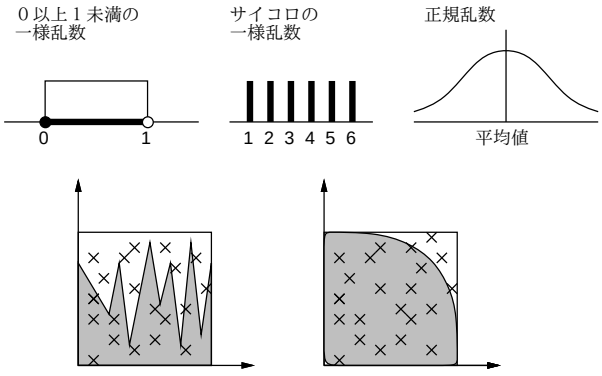
- 連立 1 次方程式の数値解法、Gauss 消去法、Gauss-Jordan、ピボット



- 反復法、Jacobi 法、Gauss-Seidel 法
- 常微分方程式の数値解法、Euler 法、Runge-Kutta 法 (2 次、4 次)



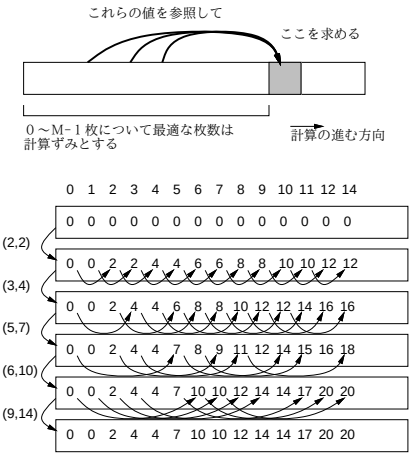
- 乱数とランダムアルゴリズム、擬似乱数、モンテカルロ法、ゲームと乱数



□ A 課題の反応: 常微分方程式をやってくれている人とゲームな人。それぞれこの程度のプログラムならやりますよという感じ? (スパルタが効いたので短いプログラムは問題なし?)

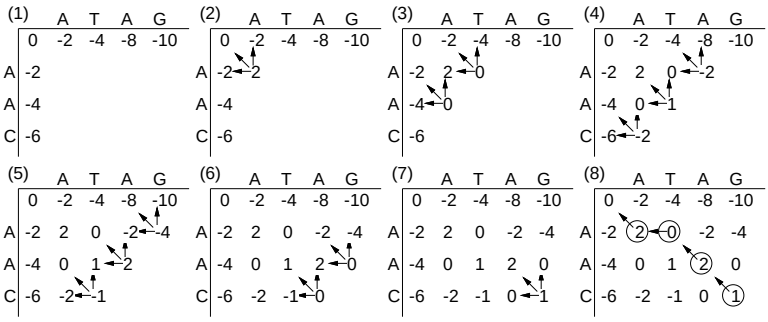
□ # 11: 動的計画法とパターン認識

- 動的計画法、コイン問題 (しらみつぶし vs DP)、ナップサック問題



- 文字列のアラインメント (原理だけ説明)

- CYK 構文解析アルゴリズム (説明は断念、上向き解析というだけ)
- パターン認識、隠れマルコフ、評価問題、複号化問題、Viterbi



□ A 課題の反応: DP が分かった人と分からない人? 分かってない人はこの短いプログラムで何が起きているのか理解困難?

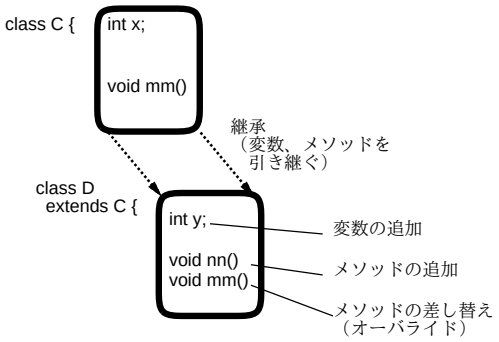
□ 久野も動的計画法は苦手なので、それを説明して分からせるのはどうしたらいいという工夫が難しい感じ

□ この後韓国出張だったので 1 回休みの回にアラインメントのプログラムを作る演習をやってもらっている。

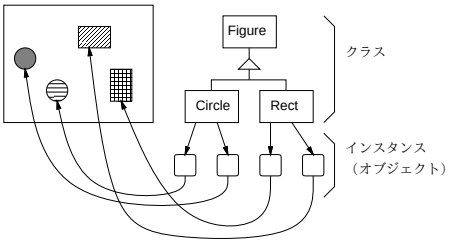
- アラインメントの計算の原理説明 (資料掲載) のみからプログラムを自力で作るという課題
- 思ったより順調にプログラムが作れていて安心した

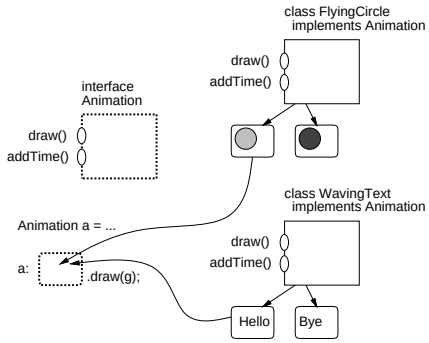
□ # 12: ウィンドウプログラムとユーザインタフェース

- 継承 (親クラス/子クラス/オーバライド)、抽象クラス、抽象メソッド

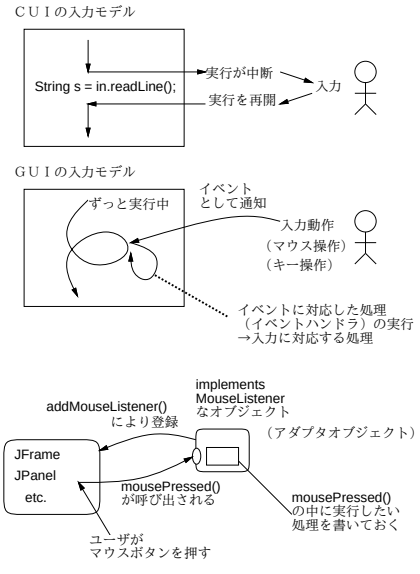


- 窓を作り出すプログラム (JFrame のサブクラス)
- スレッドとアニメーション、無名内部クラス、
- オブジェクト指向グラフィクス、インタフェースと implements

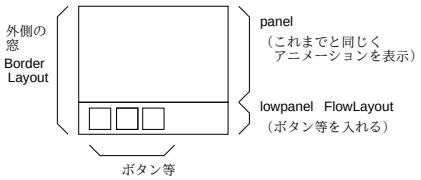


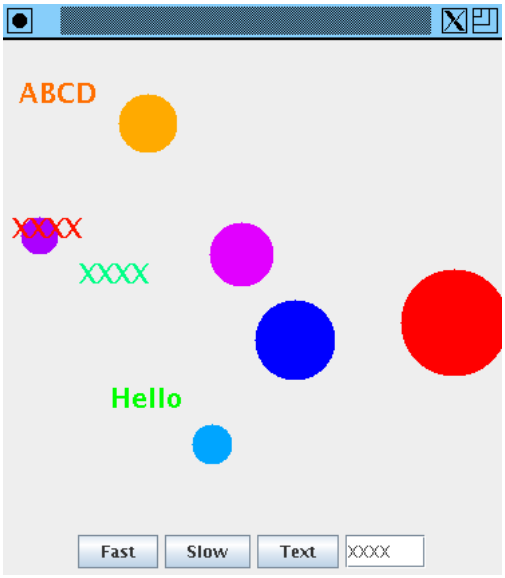


- 入力イベントの受け取り、GUI 部品



- レイアウトマネージャ、実例





☐ A 課題の反応： 最後にグラフィクスをやったので楽しかった、面白かったという感想多数 (これで終わりだから皆うれしい?)

8 まとめ

☐ 標準スライドを見て「なんてハードな」と思ったくせに、自分のカリキュラムを振り返ると輪を掛けてハードにやってました…

☐ 「CS スパルタ教育」のところであれだけやりすぎたのに好意的というのは有難いことです

☐ やっぱり最初はゴリゴリプログラム書かせて、書くのが当り前というふうにならないと理解もついて来ないと思っています

☐ 次回はスパルタのところをもっとマターリしたらいいのかどうか悩んでいます (ゆるくしたらその分うまく行くかというとそうでもないことを経験的に知っている)