

【講義メモ】 担当:平野正喜(ひらのまさき)

この講座ではプロジェクトに講義メモを書きながら進めます。この文字サイズの読める席にお座りください。

18:15~20:45(途中休憩有)。受講者数は13人です。

この講義メモは講義終了と同時に下記のサイトにPDFで掲載し、ダウンロード可能にします。ご利用ください。次回予告も掲載します。質問やコメントが送信可能です。

<https://tkuip.rundog.org>

前回の1問: スタックからの取り出しは? ア ENQ イ DEQ ウ PUSH エ POP

正解はエ。p.161 から 162 参照。

p.169 疑似言語の記述形式(続き)

・【補足】手続き:疑似言語では1つのプログラムを1つの手続きで記述したり、複数の手続きの集合で記述することがある。どちらの場合も手続き名をつけて、これを「○手続き名」で冒頭に宣言する。その後に内容を記述する。

・【補足】do-while 構文:後判定繰返しの構造。間に繰返したい内容を記述し、while の後に繰返しの継続条件を書く。do が繰返しの開始。

・【補足】while-endwhile 構文:前判定繰返しの構造。間に繰返したい内容を記述し、while の後に繰返しの継続条件を書く。endwhile が繰返しの終わり。

・【補足】if-else-endif 構文:選択の構造。If の後に条件を記述し、これが true なら(成立したら)その後の内容を実行、false なら else の後の内容を実行(else は省略可)。Endif が構造の終わり(ここで合流する)。

・【補足】if-elseif-else-endif 構文:elseif の後に条件を記述すると「if の条件が false (成立しない)ときに次の条件がチェックされる。よって「ではなくて、もしも」の意味。

・【補足】配列:疑似言語では、配列名[添字] の形式で配列の要素を示す。配列の要素は変数と同様に、代入や計算などに利用できる。

・【補足】配列の宣言:変数と同様、配列も事前に宣言が必要。要素の数を示す場合と、要素の値を列挙する場合がある。

要素の数を示す場合: 型 配列名[要素数] 例: 文字列 季節[4]

要素を列挙する場合: 型 配列名 = { 値,... } 例: 文字列 季節 = {春,夏,秋,冬}

・【補足】2次元配列:2つの添字がある配列で、行と列のイメージになる。宣言では、{ } の入れ子構造を用いる。例: 整数型 年と組の生徒数 = { {30, 25, 40}, {35, 23, 15} }

利用するには 配列名[添字, 添字] とする。例: 年と組の生徒数[1,2] で 25 が得られる

・【補足】手続きの呼出し:プログラムを構成する手続きは、外部から、または、他の手続きから呼び出されて動作する。呼び出された手続きが終わると、呼び出された位置の直後に戻る。

・【補足】引数:手続きに与えたい(手続きが受け取りたい)情報で、0個以上、いくつでも指定できる。例えば「和を表示」という手続きに引数で「1」と「2」を与えると「3」を表示、というようなこと可能。

・【補足】引数のある手続きの宣言:「○手続き名(引数の型:引数名, ...)」

例: ○和を表示 (整数型:数a, 整数型:数b)

a+b を表示する

・【補足】引数のある手続きの呼出し:「手続き名(引数の値, ...)」

例: 和を表示 (1, 2) ⇒ こうすると、引数aに1が、bに2が渡される

・【補足】引数と配列:引数として変数に加えて配列を用いることもできる

例: ○配列の合計を表示(整数型の配列:nums) …

・【補足】演算子: 計算を意味する記号や単語で、+、-、×、÷のような算術演算子、=、≠、<、>、≤、≥のような比較演算子、演算順序を優先する(、)などがある。演算順序は算数と同じで、+-より×÷が優先される。

modは剰余算で割った余りを返す(例:17 mod 4 は 1 を返す)。

and、or、not は論理演算(p.145)で 1 なら true を 0 なら false を返すので、比較演算子と一緒に使われることが多い。例: if (年齢 ≥ 20 and 年齢 ≤ 60) …

演算子「-」は左右に対象を書くと2項演算子となって引き算を行う。右だけに対象を書くと単項演算子となって符号の反転を行う。

例:「整数型 A」「A←5」「-Aを出力」とすると、-5が表示される

・【補足】return:「戻る」の意味で、手続きの途中でそれ以降をキャンセルして、呼び出し側に戻る。無駄な処理を行わないようにできる。

・【補足】構造の入れ子:if-else の中に、while-endwhile を記述したり、その中にさらにwhile-endwhile を記述するような構造(入れ子構造)が可能。

p.181 7-6-1 プログラム言語

・【補足】プログラム言語:プログラムを人間が読み書きできるように(プログラムを作成・理解しやすいように)作られた言語。コンピュータは直接理解できないので、翻訳が必要になる。用途や特性などに応じて複数のプログラム言語が使い分けられている。

・【補足】C言語:古くから使われている汎用の言語。構造が単純でチーム開発には向かない。

・【補足】C言語の改良:C言語をチーム開発、部品化、再利用に向けた言語にするためにオブジェクト指向(p.103)を導入して改良したものが、C++、Java、C#など。

・【補足】C++:C言語のオブジェクト指向対応版で、大規模システム開発に向く。高難度。

・【補足】Java:C言語のオブジェクト指向対応版で、OSに依存しない(例:Windows でもMacでも動作する)、ブラウザ上や、サーバ上など多くの場所で動作するプログラムが開発可能。安全性が高い。

・【補足】サーレット:Javaのプログラム形式の一つで、サーバ上で動作する。

・JavaVM=バーチャル(仮想)マシン(機械、ハードウェア):Javaのプログラムを動作させるための仮想的なマシンでソフトウェアでできており、各種のOSやハードウェア上で動作し、その上でJavaプログラムが動作する。OSに依存しない仕組みの基礎で、Javaプログラムからは「まるでマシンのように」見えるのがポイント。

・【補足】JavaScript:正式な仕様はECMASクリプトといい、Javaとは無関係。ブラウザ上で動作するプログラム専用で、主に動きのあるWebページの記述用。

・【補足】Python:非常に構造が小さい代わりに、機能を提供するパッケージ(ライブラリ、部品)が豊富で、特にAI系のパッケージが優秀で人工知能システムの構築などに活用されている言語。

・【補足】スクリプト言語:簡易プログラム言語のことで、実行や習得が用意だが、大規模システムや高速システムなどには向かない。JavaScript、Pythonなどがある。

・【補足】ソースプログラム:プログラム言語やスクリプト言語で書かれたプログラムのこと。人間向きで、コンピュータは直接理解できない。

・【補足】オブジェクトプログラム:ソースプログラムを翻訳した結果で、コンピュータが直接理解できる機械語になっている。なお、これに実行に必要な部品(ライブラリ)を加えた結果が

実行プログラムで、利用者などが実行できる。

・【補足】言語プロセッサ:プログラムの翻訳や実行などをサポートする仕掛けの総称。

・【補足】コンパイラ:ソースプログラムを翻訳してオブジェクトプログラムを生成する言語プロセッサ。

・【補足(参考)】リンカ:オブジェクトプログラムに必要な部品(ライブラリ)を加えて実行プログラム生成する言語プロセッサ。

・【補足】インタプリタ:逐次翻訳の意味で、ソースプログラムを1行ずつ解釈して実行する仕組み。コンパイラやリンカが不要なので手軽だが、実行速度は遅い。スクリプト言語で用いられることが多い。

・【補足】コンパイラする場合とインタプリタで実行する場合の比較:コンパイラを用いる場合、事前に翻訳する時間がかかるが、実行は速くなる。インタプリタで実行する場合、事前準備時間は無したが、実行は遅くなる。どちらが短時間かは、プログラムの規模やコンピュータの動作速度による。

p.182 7-7-1 マークアップ言語

・【補足】マークアップ言語:文字列で書かれた文書(テキスト文書)に、構造の情報や制御の情報を加えることで、各種の用途に対応する仕組みの言語。

・【補足】タグ:マークアップ言語で構造の情報や制御の情報を加えるための書式で、主に<>で囲んで記述し、通常<タグ名>で開始、</タグ名>で終了という記述にすることが多い。

・SGML=スタンダード(標準)ジェネライズド(一般化)マークアップ・ランゲージ(言語):マークアップ言語の元祖と呼ばれるもので、書籍ファイルに構造の情報を加えるなどの目的で開発された。例:<TITLE>国宝</TITLE><BODY>本文…</BODY>

・HTML=ハイパーテキスト(リンクや制御情報を持つ文書)マークアップランゲージ:SGMLを基に開発されたWebページの記述用のマークアップ言語。Webブラウザが認識する必要があるため、あらかじめ決められたタグのみを利用する(これを決めたのがW3C(p.48))。

・XML=エクステンシブル(拡張可能な)マークアップランゲージ:汎用のマークアップ言語で、記述者が自由にタグを定義できるので、EC(p.72)、EDI(p.67)などで活用。

・【補足】HTMLなどの構造:タグの開始と終了の間に、他のタグの開始と終了を記述できる。これを入れ子構造という。

例:太い<i>太くて斜め</i>ここまで太い<u>下線</u> とすると:

⇒ 太い太くて斜めここまで太い下線

・【補足(参考)】特殊なタグ:通常のタグは<タグ名>で開始タグ、</タグ名>で終了タグとなるが、HTMLの一部の特殊なタグは単独で用いるものもある。例:

第8章 コンピュータシステム

p.190 8-1-1 コンピュータの構成

・【補足(参考)】5大機能:コンピュータの5つの機能のことで、コンピュータは基本的に5つの機能で構成され、各機能の2つのルールに従って動作する。

①すべての制御は制御機能が行う。他の機能は制御はしない。

②すべての情報は記憶機能を經由する。他の機能どうして直接交換できない。

p.190 8-1-2 プロセッサ

・CPU=セントラル(中央)プロセッシング(処理)ユニット(装置):制御機能と演算機能を持つ装置・部品で、プロセッサともいう。

・マザーボード:コンピュータ内部の板状の主要部品で、多くの部品を差し込むことでコンピュ

ータを構成する。CPUなどを差し込むためのソケットを持ち、部品と部品の間の結線があらかじめ結ばれている。

・LSI＝ラージスケール(大規模)インテグレートッド(集積回路):コンピュータを構成する最小部品であるトランジスタを集積したIC＝インテグレートッド(集積)サーキット(回路)をさらに集積したもの。CPUやメモリ、制御チップなどがあり、マザーボードの上に構成される。

今日の一問: 前判定繰返しの構文は?

ア if-else-endif イ do-while ウ while-endwhile エ return

次回予告:p.190「8-1-2 プロセッサ(続き) CPUの性能を表す尺度」から