Chapter２

アルゴリズムとプログラミング

# 1. データ構造

データ構造とは、コンピュータ内にデータを格納する際の形式をいいます。データ構造は、プログラムの処理効率に大きな影響を与えるため、さまざまなデータ構造が考え出されました。

## 1. 配列

学習のポイント

✅ 添字（そえじ）によって1つのデータを指定できることを確認しよう！

✅ データの追加と削除の方法を確認しよう！

配列は、同形式の基本データ型を複数個集めたデータ構造であり、配列内の任意のデータ（要素）は、**添字**（要素番号）によって指定されます。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 添　字 | → | １ | ２ | ３ | ４ | ５ | … | ｎ |
| データ | → | Ａ | Ｂ | Ｃ | Ｄ | Ｅ | ～ | Ｚ |

配列

### １）１次元配列

上図のように、１個の添字によって任意のデータを指定するデータ構造を、１次元配列といいます。

#### ①データの追加

昇順になるように配列にデータを追加する場合、事前に格納されていたデータが消えないよう、追加する前にデータをずらす動作が必要になります。なお、手順は次のとおりです。

配列の途中にDTの内容を追加する場合

①　データの追加位置を探索する。

②　追加位置より後ろのデータを１つずつ後ろにずらす。（末尾のデータから順番に行う）

Ｔ

１

２

３

４

ｎ

ｎ＋１

Ａ

Ｂ

Ｄ

Ｅ

…

Ｚ

①

②

③

④

DT：追加するデータ

ｎ：データ数

③　データ（ＤＴ）を配列に書き込む。

Ｙ

１

２

３

４

ｎ

ｎ＋１

Ａ

Ｂ

C

Ｄ

…

Ｔ

Ｚ

Ｃ

DT

④　データ数ｎに１を加算する。

配列へのデータの追加（例）

#### ②データの削除

配列からデータを削除する場合、削除位置よりも後方に格納されているデータを前にずらすことによって行われます。

配列からDTの内容と等しいデータを削除する場合

①　削除するデータを探索する。

②　削除するデータより後ろのデータを１つずつ前にずらす。（配列の先頭に近いデータから順番に行う）

DT：削除するデータ

ｎ：データ数

Ｃ

DT

Ｔ

１

２

３

４

ｎ－１

ｎ

Ａ

Ｂ

Ｃ

Ｄ

…

Ｙ

①

②

③

④

Ｚ

③　データ数ｎから１を引く。

物理的な削除（例）

### ２）２次元配列

２次元配列は、行列（matrix）または表ともいい、２つの次元数ｍ、ｎをもっています。ｍは行列の行を、ｎは行列の列を表し、ｍ行×ｎ列で表現します。行列内の各データの位置は、２つの添字によって指定します。

１

２

３

４

５

６

１

２

３

４

５

６

列

行

配列ａ

ａ(3,4)

２次元配列

|  |
| --- |
| 例題  データ構造の特徴のうち，適切なものはどれか。  ア　配列は，添字によってデータを任意の順序で読み出すことができる。  イ　配列を用いることによって，データ構造とアルゴリズムを独立させることができる。  ウ　リストは，添字によってデータの検索や更新ができる。  エ　リストは，データの挿入や削除のときに既存のデータを移動する必要がある。  配列は、同形式のデータを複数個集めたデータ構造であり、添字（配列の何番目に配置されているかを表す数値）で指定することによって、配列内の任意のデータを読み出すことができます。  なお、リストは、次のデータの格納位置を、ポインタと呼ばれる領域で表すデータ構造です。  初級システムアドミニストレータ　平成19年度春　問34　[出題頻度：★★☆]  解答－ア |

別冊演習ドリル 》 1-95,96

**１-９１～１０３**

解答－ア

## 2. リスト

学習のポイント

✅ ポインタの役割をおさえ、構造を覚えよう！

✅ データの追加と削除の方法を理解しよう！

リストは、次のデータの格納位置を、**ポインタ**と呼ばれる領域で表すデータ構造です。

最初のデータ

次のデータ

・

Ｂ

・

Ａ

リスト

|  |  |  |
| --- | --- | --- |
| データ構造 | データアクセス方法 | データの挿入・削除 |
| 配 列 | 配列名と添字を指定することでデータへ直接アクセスすることができる。 データを読み取るための作業は一定。 | 挿入・削除をする場所より後ろのデータは１つずつずらさなければならない。 **挿入・削除をする場所によってずらす作業量が異なる。** |
| リスト | リストの先頭から順番にたどっていく必要がある。目的のデータを探索するため、探索する作業量が一定ではない。 | 挿入・削除する場所のポインタを書き換えるだけで挿入・削除ができる。 **挿入・削除の場所がどこでも作業量が一定となる。** |

配列とリストの違い

リストには、次の種類があります。

### １）単方向リスト（one-way list）

単方向リストは、１つの方向だけポインタを連結したリストであり、図の単方向リストであれば「Ａ→Ｂ→Ｃ」の順番でのみ探索することができます。

ルート□

Ａ

・

最初

Ｂ

・

中間

Ｃ

×

最後

×：リストの最後を示す

□：最初のデータへのポインタ（ルート）

単方向リスト（例）

#### ①データの探索

単方向リストからデータを探索する場合、先頭データの位置（ルート）から順番に、ポインタに従って行われます。

①　ルートが指すアドレス10の位置に格納されているデータＡと比較。

②　データＡのポインタが指すアドレス25の位置に格納されているデータＢと比較

③　データＢのポインタが指すアドレス32の位置に格納されているデータＣと比較

④　データＣのポインタが指すアドレス40の位置に格納されているデータＤと比較

⑤　データＤのポインタには末尾を表す×が格納されているため、一致するデータはなし

（ルート）

Ａ

25

10

10

Ｂ

32

25

Ｃ

40

32

Ｄ

×

40

データのアドレス

単方向リストの探索（例）

#### ②データの追加

単方向リストへデータを追加する場合、配列のように格納場所を移動するのではなく、ポインタを付け替えることによって行います。

データＢをデータＡとデータＣの間に追加する場合

①　データＢの追加位置を探索する。

②　データＢを記憶装置に書き込む。

③　データＢのポインタがデータＣを指すように変更する。  
　　（データＡのポインタをデータＢのポインタに移す。）

④　データＡのポインタがデータＢを指すように変更する。  
　　（データＢの格納場所をデータＡのポインタに書き込む。）

Ａ

32

25

10

Ｂ

32

25

Ｃ

32

単方向リストへのデータの追加（例）

#### ③データの削除

単方向リストからデータを削除する場合も、追加のときと同様にポインタを付け替えることによって行います。

データＢを削除する場合

①　削除データの格納位置を探索する。

②　データＡのポインタがデータＣを指すように変更する。（データＢのポインタをデータＡのポインタに移す。）

Ａ

25

32

10

Ｂ

32

25

Ｃ

32

単方向リストのデータの削除（例）

### ２）双方向リスト（two-way list）

双方向リストは、２つのポインタを付けることにより、両方向からの探索を可能にしたリストです。図の双方向リストであれば「Ａ→Ｂ→Ｃ」の順番だけでなく、「Ｃ→Ｂ→Ａ」の順番でも探索が可能となります。

Ａ

×

先頭

・

Ｂ

・

・

前への

ポインタ

データ

後への

ポインタ

Ｃ

・

末尾

×

双方向リスト（例）

#### ①データの探索

双方向リストでは、両方からの探索が可能ですが、通常、一方のポインタを使って、単方向リストと同様に行います。

#### ②データの追加

双方向リストへデータを追加する場合も、ポインタを付け替えることによって行います。

データＢをデータＡとデータＣの間に追加する場合

①　データＢの追加位置を探索する。

②　データＢを記憶装置に書き込む。

③　データＢの後ろへのポインタがデータＣを指すように変更する。  
　　（データＡの後ろへのポインタの内容をデータＢの後ろへのポインタに移す。）

④　データＢの前へのポインタがデータＡを指すように変更する。  
　　（データＣの前へのポインタの内容をデータＢの前へのポインタに移す。）

⑤　データＡの後ろへのポインタがデータＢを指すように変更する。  
　　（データＢの格納位置をデータＡの後ろへのポインタに書き込む。）

⑥　データＣの前へのポインタがデータＢを指すように変更する。  
　　（データＢの格納位置をデータＣの前へのポインタに書き込む。）

Ａ

10

32

25

Ｃ

10

25

Ｂ

10

32

**25**

32

双方向リストへのデータの追加（例）

#### ③データの削除

双方向リストからデータを削除する場合も、ポインタを付け替えることによって行います。

データＢを削除する場合

①　削除データの格納位置を探索する。

②　データＡの後ろへのポインタがデータＣを指すように変更する。  
　　（データＢの後ろへのポインタの内容をデータＡの後ろへのポインタに移す。）

③　データＣの前へのポインタがデータＡを指すように変更する。  
　　（データＢの前へのポインタの内容をデータＣの前へのポインタに移す。）

32

Ａ

10

25

32

Ｃ

Ｂ

10

32

25

25

10

双方向リストのデータの削除（例）

🏋プラスアルファ

**●環状リスト**（circular list）

環状リストは、リストの末尾のデータのポインタに、先頭のデータの格納位置を格納することによって、再び先頭からの探索を可能にしたリストです。

Ａ

Ｂ

Ｃ

先頭

・

・

・

末尾

ルート□

環状リスト（例）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題　🏋プラスアルファ  図のような単方向リストがある。“トウキョウ”がリストの先頭であり，そのポインタには次に続くデータのアドレスが入っている。また，“ミラノ”はリストの最後であり，そのポインタには０が入っている。  “ロンドン”を“パリ”に置き換えることができる処理はどれか。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 先頭データへの  ポインタ | | アドレス | データ部分 | ポインタ | | 120 | 100 | ウィーン | 160 | |  | 120 | トウキョウ | 180 | |  | 140 | パリ | 999 | |  | 160 | ミラノ | 0 | |  | 180 | ロンドン | 100 |   ア　パリのポインタを100とし，トウキョウのポインタを140とする。  イ　パリのポインタを100とし，ロンドンのポインタを140とする。  ウ　パリのポインタを180とし，トウキョウのポインタを140とする。  エ　パリのポインタを180とし，ロンドンのポインタを140とする。  置き換え前の連結イメージを表すと次のようになります。    連結イメージを作成してみると、データとして存在しているパリは、リストには連結されていないことが分かります。このように、必ずしも存在しているデータがすべてリストに連結されているとは限りません。  ソフトウェア開発　平成13年度春　問10　[出題頻度：★★★]  解答－ア |

別冊演習ドリル 》 1-97～101

## 3. スタックとキュー

学習のポイント

✅ それぞれの構造におけるデータの入力順と出力順の関係を覚えよう！

### １）スタック

スタックは、一方の端だけを使用してデータの出し入れを行うデータ構造であり、データ（要素）を格納する操作を**push**、データを取り出す操作を**pop**といいます。

Ｃ

Ｂ

Ａ

データ

データの格納

（push）

データ

頂上（Top）

底（Bottom）（Bottom）

データの取出し

（pop）

スタックの概念

スタックは、後入れ先出し方式（**LIFO**：Last In First Out）のデータ構造であり、最後に格納したものが最初に取り出されます。したがって、上図では、Ａ→Ｂ→Ｃの順にpushされ、Ｃ→Ｂ→Ａの順にpopされます。

|  |
| --- |
| 例題  次の二つのスタック操作を定義する。  PUSH ｎ：スタックにデータ（整数値ｎ）をプッシュする。  POP 　 ：スタックからデータをポップする。  空のスタックに対して，次の順序でスタック操作を行った結果はどれか。  PUSH 1 → PUSH 5 → POP → PUSH 7 → PUSH 6 → PUSH 4 → POP → POP → PUSH 3  ア イ ウ エ  ６  ４  ３  ３  ７  １  ３  ４  ６  １  ７  ３  スタックとは、一方の端だけを使用してデータの出し入れを行うデータ構造です。後入れ先出し方式のデータ構造であり、最後に格納したものが最初に取り出されます。  １  PUSH　１  PUSH　５  ５  １  POP  １  PUSH　7  ７  １  PUSH　６  ６  ７  １  ４  ６  ７  １  PUSH　４  POP  ６  ７  １  POP  ７  １  PUSH　３  ３  ７  １  基本情報　平成30年度春　問5　[出題頻度：★★★]  解答－ウ |

別冊演習ドリル 》 1-102～111

### ２）キュー

キューは、待ち行列とも呼ばれ、一方の端からのみデータを格納し、取り出しはもう一方の端からのみ行うデータ構造です。要素を格納する操作をenqueue（エンキュー）、要素を取り出す操作をdequeue（デキュー）といいます。

Ｃ

Ｂ

Ａ

データ

データの格納

（enqueue）

末尾

先頭

新しい

古い

データ

データの取出し

（dequeue）

キューの概念

キューは、先入れ先出し方式（**FIFO**：First In First Out）のデータ構造であり、最初に格納したものが最初に取り出されます。したがって、上図では、Ａ→Ｂ→Ｃの順にenqueueされ、Ａ→Ｂ→Ｃの順にdequeueされます。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  待ち行列に対する操作を，次のとおり定義する。  ENQ　 n ：待ち行列にデータ n を挿入する。  DEQ ：待ち行列からデータを取り出す。  空の待ち行列に対し， ENQ 1，ENQ 2，ENQ 3，DEQ，ENQ 4，ENQ 5，DEQ，ENQ 6，DEQ，DEQ の操作を行った。次に DEQ操作を行ったとき，取り出されるデータはどれか。  ア　１ イ　２ ウ　５ エ　６  与えられた操作を実行し、更にDEQを行った場合の待ち行列の状態を表すと、次のようになります。なお、待ち行列のデータの挿入・取り出しは次のように行うものとします。  待ち行列   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ENQ　→ |  |  |  |  | →　DEQ |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 手続き | ENQ | 待ち行列 | | | | DEQ | | ENQ　１ | １　→ |  |  |  | １ |  | | ENQ　２ | ２　→ |  |  | ２ | １ |  | | ENQ　３ | ３　→ |  | ３ | ２ | １ |  | | DEQ |  |  |  | ３ | ２ | →　１ | | ENQ　４ | ４　→ |  | ４ | ３ | ２ |  | | ENQ　５ | ５　→ | ５ | ４ | ３ | ２ |  | | DEQ |  |  | ５ | ４ | ３ | →　２ | | ENQ　６ | ６　→ | ６ | ５ | ４ | ３ |  | | DEQ |  |  | ６ | ５ | ４ | →　３ | | DEQ |  |  |  | ６ | ５ | →　４ | | **DEQ** |  |  |  |  | ６ | **→　５** |   基本情報　平成30年度秋　問5　[出題頻度：★★★]  解答－ウ |

別冊演習ドリル 》 1-112～114

## 4. 木構造

学習のポイント

✅ 木構造を構成する各要素（根、節など）の名称を覚えよう！

✅ ２分探索木の定義を覚えよう！

木構造とは、要素同士の階層関係を表現するデータ構造であり、データ間の分岐関係を表現するのに適しています。

|  |  |
| --- | --- |
| 節（ノード） | データを格納する領域（左図の①～⑩） |
| 枝（辺） | 節を結ぶ線 |
| 根 | 一番上に位置する節（左図の①） |
| 葉 | 一番下に位置し，枝の分岐していない節 （左図の④，⑥，⑧～⑩） |

１

２

４

５

６

８

９

３

７

10

根

節

葉

枝

木（例）

図中の節①と節②、③は階層関係が成立していますが、節①のことを親、節②、③のことを子といい、「節①と節②または節③は親子関係にある」という表現をします。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  節点１，２，…，ｎをもつ木を表現するために，大きさｎの整数型配列Ａ[１]，Ａ[２]，…，Ａ[ｎ]を用意して，節点ｉの親の番号をＡ[ｉ]に格納する。節点ｋが根の場合はＡ[ｋ]＝０とする。表に示す配列が表す木の葉の数は，幾つか。   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | ｉ | １ | ２ | ３ | ４ | ５ | ６ | ７ | ８ | | A[ i ] | ０ | １ | 1 | ３ | ３ | ５ | ５ | ５ |   ア　１ イ　３ ウ　５ エ　７  問題に定義されている配列を木構造で表現すると次のようになります。  １  ２  ３  ４  ５  ６  ８  ７  葉とは子をもたない節であるため、葉の数は５となります。  基本情報　平成22年度秋　問6　[出題頻度：★☆☆]  解答－ウ |

別冊演習ドリル 》 1-115

### １）２分木（binary tree）

２分木は、各節の子の数が２つ以下の木のことであり、２つの子をそれぞれ左の子、右の子と区別します。また、左の子を根とする部分を左部分木、右の子を根とする部分を右部分木といいます。

節①の左部分木

１

２

４

５

７

８

３

６

節①の右部分木

２分木（例）

### ２）完全２分木

完全２分木は、２分木のうち、根から葉までの経路の長さが等しい２分木のことです。完全２分木が成立するのは、厳密には節の数が２ｈ＋１－１個（２分木の深さをｈとした場合）のときですが、常に節の数が２ｈ＋１－１個であるとは限らないため、根から葉までの各経路の長さの差が１以下であり、かつ一番下は左から順に葉が詰められている２分木を、完全２分木と定義しています。

１

２

４

５

６

３

７

４

２

５

８

６

７

３

１

完全２分木（例）

|  |
| --- |
| 例題  すべての葉が同じ深さをもち，葉以外のすべての節点が二つの子をもつ２分木に関して，節点数と深さの関係を表す式はどれか。ここで，*ｎ*は節点数，*ｋ*は根から葉までの深さを表す。例に示す２分木の深さ*ｋ*は２である。  例  *ｋ*＝２  ア　*ｎ*＝*ｋ* (*ｋ*＋１)＋１ イ　*ｎ*＝２*ｋ*＋３  ウ　*ｎ*＝２*ｋ*＋１－１ エ　*ｎ*＝(*ｋ*－１) (*ｋ*＋１)＋４  深さ（*ｋ*）が３の２分木で確かめていくと答えを導き出すことができます。深さ（*ｋ*）が３の２分木は、節点数が15になります。（次の図を参照）  *ｋ*＝３  これを各選択肢の式に当てはめて、*ｎ*＝15になる選択肢を探します。  ア　ｎ＝３(３＋１)＋１ イ　ｎ＝２３＋３  ＝12＋１ ＝８＋３  ＝13 ＝11  ウ　ｎ＝２３＋１－１ エ　ｎ＝(３－１)(３＋１)＋４  ＝16－1 ＝２×４＋４  ＝15 ＝12  以上の計算の結果、適切な式はウであることが分かります。  基本情報　平成17年度秋　問12　[出題頻度：★☆☆]  解答－ウ |

別冊演習ドリル 》 1-116

### ３）２分探索木（binary search tree）

２分探索木は、各節がキーをもつ２分木であり、左部分木の全ての節のキー値は親のキー値より小さく、右部分木の全ての節のキー値は親と等しいか、親よりも大きい木のことです。つまり、「左部分木のキー値＜親のキー値≦右部分木のキー値」の関係が成り立つ木を、２分探索木といいます。

１

２

４

３

９

７

６

８

５

２分探索木（例）

|  |
| --- |
| 例題  空の２分探索木に７個の整数13，17，15，８，３，10，20をこの順番に挿入した後，この２分探索木から２個の整数８，３をこの順番に削除した結果はどれか。  ア イ  17  13  20  15  10  15  10  13  17  20  15  13  17  10  20  10  13  17  15  20  ウ エ  問題文にある７個の整数13、17、15、８、３、10、20を空の２分探索木に挿入すると、図のようになります。  13  ８  20  17  ３  10  15  この２分探索木から整数８を削除すると、左の図のようになります。さらに整数３を削除すると右の図のようになります。  13  ３  17  10  15  20  13  10  17  15  20  ソフトウェア開発　平成14年度春　問9　[出題頻度：★★★]  解答－イ |

別冊演習ドリル 》 1-117～121

🏋プラスアルファ

**●**Ｂ木

木を作り直し、根から葉までの高さを一定に保つような木のことを、**バランス木**といいます。

Ｂ木（B-tree）は、バランス木の一種で、葉だけが値をもち、葉以外の節は値をもたずキーだけをもっています。  
キーは、ポインタで示す節や、その子のもつ値の範囲を示します。

④

①

②

11

６

７

⑨

⑦

⑩

⑬

⑪

⑮

⑭

⑰

⑳

⑱

14

18

⑥

データ

Ｂ木（例）

|  |
| --- |
| 例題　🏋プラスアルファ  データ構造に関する記述のうち，Ｂ木の説明として適切なものはどれか。  ア　ある特定のアルゴリズムに従って，レコードのキー値から物理的な格納アドレスを求めてレコードを格納する。  イ　索引部の各ノードのキー値を中心にして，小さい側のレコード数と大きい側のレコード数の比率が，ある許容範囲内に収まるように動的に再配置しながら格納する。  ウ　レコードの物理的配置とは独立に，論理的にレコードをつなぐポインタによって，レコードを関係づけて格納する。  エ　レコードをキー値の昇順にしてトラックなどのアクセス単位（ページ）ごとに格納し，各ページ内の最大キー値とそのページの番地をもつ索引を作る。  Ｂ木は、バランス木の一種で、データ検索回数のばらつきをなくすため、追加、削除があるたびに再編成を行い、平衡を保つようにするもので、根から葉までの深さが等しくなっています。Ｂ木は、データベースのデータ構造に用いられています。  ア　物理的な格納アドレスを求めるのは、ハッシングのアルゴリズムです。Ｂ木では、論理アドレスを求めます。  ウ　リスト構造に関する記述です。  エ　索引付き配列に関する記述です。  ソフトウェア開発　平成18年度春　問9　[出題頻度：★☆☆]  解答－イ |

別冊演習ドリル 》 1-122

# 2. アルゴリズム

アルゴリズムとは、問題を解くための処理手順を定式化したものです。コンピュータで処理を行う際にも、まず対象となる問題のアルゴリズムを記述することから始めます。

## 1. 流れ図

学習のポイント

✅ 流れ図の見方が分かるようになろう！

流れ図は、アルゴリズムの記述法としては最も古い歴史をもち、JIS（日本産業規格）にも定義されている方法です。あらかじめ決められている記号を処理の流れに従って組み合わせることで、アルゴリズムを表現します。流れ図に使われる記号には次のものがあります。

|  |  |  |
| --- | --- | --- |
| 記　号 | 名　称 | 説　明 |
|  | 端子 | 流れ図の開始と終了を表す記号 |
|  | 処理 | 実行する処理を表す記号 |
|  | 判断 | 条件によって次に行う処理を選択するという処理を表す記号 |
|  | 線 | 処理の流れを表す |
|  | ループ始端 | 繰り返しの始まりを表す記号 |
|  | ループ終端 | 繰り返しの終わりを表す記号 |

流れ図記号（例）

流れ図は、これらの記号中に具体的な処理内容を簡潔に記述します。また、制御の流れは、実線を用いて表し、原則として左から右、上から下へ流れるように記述します。原則とは逆の方向に制御が流れる場合には、矢印を用います。

アルゴリズムの基本的な構造と流れ図をまとめると、次のようになります。

### ①順次構造

各計算や操作が直線的につながっている構造です。

### ②選択構造

条件により処理内容が分かれる構造です。

### ③繰返し構造

終了条件を満たすまで（または繰返し条件を満たしている間）、一連の処理を繰り返すといいう構造です。

順次構造 選択構造 繰返し構造

処理１

処理２

処理３

処理１

処理２

真

偽

条件

処理

ループ名

終了条件

（又は継続条件）

ループ名

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  次の流れ図において，① → ② → ③ → ⑤ → ② → ③ → ④ → ② → ⑥　の順に実行させるために，①においてmとnに与えるべき初期値aとbの関係はどれか。ここで，a，bはともに正の整数とする。  ①  ②  m ← a  ｎ ← ｂ  終了  開始  mの値を印字  m：n  ＝  ≠  ＞  ＜  ③  ⑤  ⑥  ④  m ← (m－n)  n ← (n－m)  m：n  ア　a＝2b イ　2a＝b ウ　2a＝3b エ　3a＝2b  流れ図をトレースし、m、nの値の変化を考えます。それぞれの分岐条件とm、nの値を一覧にすると次のようになります。   |  |  |  |  | | --- | --- | --- | --- | |  | 分岐 | ｍ | ｎ | | ② | ｍ≠ｎ | ａ | ｂ | | ③ | ｍ＜ｎ | ａ | ｂ | | ⑤ | ｎ←(ｎ－ｍ) | ａ | ｂ－ａ | | ② | ｍ≠ｎ | ａ | ｂ－ａ | | ③ | ｍ＞ｎ | ａ | ｂ－ａ | | ④ | ｍ←(ｍ－ｎ) | ２ａ－ｂ | ｂ－ａ | | ② | ｍ＝ｎ |  |  |   最後の②ではｍ＝ｎが成り立つので、ｍ、ｎにそれぞれ値を代入すると、  2ａ－ｂ＝ｂ－ａ  ∴ 3ａ＝2ｂ  これが、初期値の条件となります。  基本情報　令和5年度公開　問11　[出題頻度：★★★]  解答－エ |

別冊演習ドリル 》 1-123～129

## 2. 代表的なアルゴリズム

学習のポイント

✅ 代表的な整列方法について、それぞれの名称と手法、比較回数（計算量）を覚えよう！

✅ 代表的な探索方法について、それぞれの名称と長所、短所、比較回数を覚えよう！

### １）整列（sort）

データ列をある規則に従って並び替えることを整列といい、その処理手順を総称して、ソートアルゴリズムと呼びます。なお小さい順に並べることを**昇順**、大きい順に並べることを**降順**といいます。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 40  ①  ②  ③  ④ | 20 | 50 | 30 | 10 |
|  |  |  |  |  |
| 20  ①  ②  ③ | 40 | 30 | 10 | 50 |
|  |  |  |  |  |
| 20  ①  ② | 30 | 10 | 40 | 50 |
|  |  |  |  |  |
| 20 | 10 | 30 | 40 | 50 |
|  |  |  |  |  |
| 10 | 20  ① | 30 | 40 | 50 |

#### ①バブルソート

バブルソートは、データ列の端から順番に隣り合ったデータ同士を比較し、順番が逆ならば交換する、という操作を繰り返すことによって整列を行う方法です。

データ数をｎとすると、比較回数はｎ(ｎ－１)／２回、計算量はｎ２に比例（*O*（ｎ２））します。最小比較回数はｎ－１回となります。

なお、***O*（オーダ）**とはおおよその量を表すときに使う単位であり、ｎの増加に伴い、計算量がおもに何に従って増加するかを表します。

※網掛け箇所のデータから順番に完成する

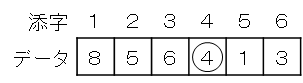
バブルソートの比較のイメージ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  データ列の隣り合う要素の値を比較し，小さい方が右にあれば交換する。この操作をデータ列の左端から右端まで繰り返す処理を１回のパスとする。  次のデータ列でパスを２回繰り返した後のデータ列の内容を示しているものはどれか。   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ５ | ４ | １ | ３ | ６ | ２ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | ア |  | １ | ３ | ２ | ４ | ５ | ６ |  | イ |  | １ | ３ | ４ | ２ | ５ | ６ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | ウ |  | ４ | １ | ５ | ３ | ２ | ６ |  | エ |  | ４ | １ | ５ | ３ | ６ | ２ |   隣り合う要素を、配列の先頭から順に比較していくバブルソートのアルゴリズムです。１回目のパスが終了すると、配列の最後に最大値が格納されます。また、２回目のパスにより、２番目に大きい値が最後から２番目の要素に格納されます。問題に与えられたデータ列を、２回目のパスまでトレースすると、次のようになります。  １回目のパス ２回目のパス  ５４１３６２ ４１３５２６  ４５１３６２ １４３５２６  ４１５３６２ １３４５２６  ４１３５６２ １３４５２６  ４１３５６２ １３４２５６  ４１３５２６  二種　平成12年度秋　問15　[出題頻度：★★☆]  解答－イ |

別冊演習ドリル 》 1-130,131

#### ②クイックソート

クイックソートは、ある大きさの問題を小さく分割し、効率よく問題を解決するという**分割統治法**を適用したソートアルゴリズムです。具体的には、整列の対象となるデータを、ある値より小さなデータからなるグループと、ある値より大きなデータからなるグループに分割し、この操作を、各グループともデータ数が１になるまで繰り返すことにより整列を行う方法です。なお、データ数をｎとすると平均計算量は*O*（ｎlog2ｎ）、最大の計算量は*O*（ｎ２）となります。



ａ：④を軸とし、軸より小さいものは前半へ、

大きいものは後半へ移動する。

ｄ：整列完了。

ｂ：軸より小さい部分データ列に対して、

③を軸として再分割をする。

ｃ：軸より大きい部分データ列に対して、

⑥を軸として再分割をする。

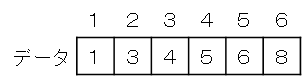












クイックソートの手順（例）

|  |
| --- |
| 例題  クイックソートの処理方法を説明したものはどれか。  ア　既に整列済みのデータ列の正しい位置に，データを追加する操作を繰り返していく方法である。  イ　データ中の最小値を求め，次にそれを除いた部分の中から最小値を求める。この操作を繰り返していく方法である。  ウ　適当な基準値を選び，それよりも小さな値のグループと大きな値のグループにデータを分割する。同様にして，グループの中で基準値を選び，それぞれのグループを分割する。この操作を繰り返していく方法である。  エ　隣り合ったデータの比較と入替えを繰り返すことによって，小さな値のデータを次第に端の方に移していく方法である。  クイックソートは、次のような手順で行います。  まず、ある基準となる要素を決めて、その要素より小さな値と大きな値で配列を２つに分けます。そして、それぞれの配列で、また同じように基準値を決め、大小関係により配列を２つに分けます。この作業を、これ以上細分化できないところまで繰り返せば、もとの配列の要素は整列されたことになります。  ア　挿入ソートに関する記述です。  イ　選択ソートに関する記述です。  エ　バブルソートに関する記述です。  基本情報　平成30年度秋　問6　[出題頻度：★☆☆]  解答－ウ |

別冊演習ドリル 》 1-132

#### ③選択ソート

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 20  ①  ②  ③  ④ | 10 | 50 | 40 | 30 |
|  |  |  |  |  |
| 10 | 20  ①  ②  ③ | 50 | 40 | 30 |
|  |  |  |  |  |
| 10 | 20 | 50  ①  ② | 40 | 30 |
|  |  |  |  |  |
| 10 | 20 | 30 | 50  ① | 40 |
|  |  |  |  |  |
| 10 | 20 | 30 | 40 | 50 |

選択ソートは、データの中で最小値（または最大値）を探して配列の端から順に格納する動作を繰り返すことによって整列を行う方法です。

なお、データ数をｎとすると、比較回数はｎ(ｎ－１)／２回、計算量はｎ２に比例（*O*（ｎ２））します。

選択ソートの比較のイメージ

|  |
| --- |
| 例題  未整列の配列*A* [ *i* ]（*i* ＝1，2，... ，*n*）を，次のアルゴリズムで整列する。要素同士の比較回数のオーダを表す式はどれか。  〔アルゴリズム〕  (1)　*A* [1]～*A* [*n*] の中から最小の要素を探し，それを*A* [1] と交換する。  (2)　*A* [2]～*A* [*n*] の中から最小の要素を探し，それを*A* [2] と交換する。  (3)　同様に，範囲を狭めながら処理を繰り返す。  ア　*O* (log2 *n* ) イ　*O* (*n* ) ウ　*O* (*n* log2 *n* ) エ　*O* ( *n*2 )  対象となる範囲から最小値を選んで整列を進めていくこのアルゴリズムは、選択ソートです。ソートアルゴリズムの中で、バブルソート・選択ソート・挿入ソートの３つを併せて整列の基本三法と呼ぶことがありますが、要素同士の比較回数を表す計算量はいずれもｎ２です。計算量は一般に*О*記法（オーダ記法）と呼ばれる表記法を用いて表現されます。  基本情報　平成14年度秋　問13　[出題頻度：★☆☆]  解答－エ |

別冊演習ドリル 》 1-133

#### ④ヒープソート

ヒープソートは、ヒープと呼ばれる木を用いた整列法です。整列の対象となる要素からヒープを構成し、ヒープの親子関係を正していくことによって整列を行う方法です。ヒープの特徴は、全ての節が必ず親＞子（昇順の場合）または子＞親（降順の場合）となる構造をしていることです。その結果、根が全ての節点の中で最大の値（昇順の場合）または最小の値（降順の場合）となります。なお、データ数をｎとすると計算量は*O*（ｎlog2ｎ）となります。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ３ | ２ | ６ | ７ | ４ | ５ | １ | を昇順に整列する場合 |

１．ヒープを作成する。

配列のデータをもとに木を作成し、親＞子となるように親子を入れ替える。

その結果、根には必ず最大値が来る。

７

４

５

１

２

６

３

７

４

５

１

２

６

３

２

４

５

１

７

６

３

７

４

５

１

２

６

３

２

４

５

１

３

６

７

２

３

５

１

４

６

７

２．整列を行う。

具体的には、未整列の部分を順序木にして、そこから最大値（昇順の場合）または最小値（降順の場合）を取出して、すでに整列された部分に移す操作を繰り返して未整列部分を縮めていく。

７

６

５

４

３

１

２

２

３

１

４

７

６

５

２

３

１

４

７

６

５

１

２

７

６

５

４

３

３

１

２

７

６

５

４

２

１

３

７

６

５

４

２

１

７

６

５

４

３

２

３

４

１

５

７

６

２

３

４

５

１

７

６

２

４

１

３

７

６

５

２

３

５

４

１

６

７

２

３

５

７

４

６

１

２

３

５

１

４

６

７

２

３

１

４

５

６

７

は整列済みのデータ

ヒープソートの手順（例）

|  |
| --- |
| 例題  親の節の値が子の節の値より小さいヒープがある。このヒープへの挿入は，要素を最後尾に追加し，その要素が親よりも小さい間，親と子を交換することを繰り返せばよい。次のヒープの＊の位置に要素７を追加したとき，Ａの位置に来る要素はどれか。  25  11  24  34  29  14  28  19  ９  ＊  Ａ  ア　７ イ　11 ウ　24 エ　25  このヒープは「親の節の値が子の節の値より小さいヒープ」という条件であるため、新しい要素を挿入した後に、条件が満たされるまで交換処理を繰り返すことになります。  14  28  19  ９  25  11  24  34  29  ７  14  28  19  ９  ７  11  24  34  29  25  14  28  19  ９  11  ７  24  34  29  25  14  28  19  ７  11  ９  24  34  29  25  親(25)＞子(7)  なので入れ替える  親(11)＞子(7)  なので入れ替える  親(9)＞子(7)  なので入れ替える  したがって、Ａ（二重丸）の位置は11となります。  基本情報　平成20年度秋　問12　[出題頻度：★☆☆]  解答－イ |

別冊演習ドリル 》 1-134,135

#### ⑤挿入ソート

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 40  ① | 20 | 10 | 50 | 30 |
|  |  |  |  |  |
| 20 | 40 | 10 | 50 | 30 |
|  |  |  |  |  |
| 10 | 20 | 40  ①  ②  ① | 50 | 30 |
|  |  |  |  |  |
| 10 | 20  ①  ②  ③ | 40 | 50 | 30 |
|  |  |  |  |  |
| 10 | 20 | 30 | 40 | 50 |

挿入ソートは、データを部分的に整列していき、整列されている列の中に未整列のデータを追加することで整列を行う方法です。そのため、最初のデータの並び方によって比較回数が異なります。

データ数をｎとすると、平均比較回数は約ｎ(ｎ－１)／４回となり、計算量は*O*（ｎ２）になります。

挿入ソートの比較のイメージ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  四つの数の並び（4，1，3，2）を，ある整列アルゴリズムに従って昇順に並べ替えたところ，数の入替えは次のとおり行われた。この整列アルゴリズムはどれか。  （1，4，3，2）  （1，3，4，2）  （1，2，3，4）  ア　クイックソート イ　選択ソート  ウ　挿入ソート エ　バブルソート  挿入ソートによる整列アルゴリズムは次のとおりであり、網掛けの箇所で比較します。   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | １回目の入替え | ４ | １ | ３ | ２ | １を４の前に挿入 | | **１** | **４** | **３** | **２** | １と４が整列済み | | ２回目の入替え | １ | ４ | ３ | ２ | 整列済みの範囲に対して、  ３を適切な位置に挿入する | | １ | ３ | ４ | ２ | | **１** | **３** | **４** | **２** | １、３、４が整列済み | | ３回目の入替え | １ | ３ | ４ | ２ | 整列済みの範囲に対して、  ２を適切な位置に挿入する | | １ | ３ | ２ | ４ | | １ | ２ | ３ | ４ | | **１** | **２** | **３** | **４** | 全てが整列済み |   基本情報　平成14年度春　問14　[出題頻度：★☆☆]  解答－ウ |

🏋プラスアルファ

**●シェルソート**

シェルソートは、挿入ソートを改良した方法であり、一定間隔（通常は要素数の１／２）離れた要素同士で挿入ソートを行い、その後、間隔を半分にしながら挿入ソートを繰り返すものです。データ数をｎとすると、計算量は*O*（ｎ３/２）となります。

|  |
| --- |
| 例題　🏋プラスアルファ  次の手順はシェルソートによる整列を示している。データ列７，２，８，３，１，９，４，５，６を手順(1)～(4)に従って整列するとき，手順(3)を何回繰り返して完了するか。ここで，[　] は小数点以下を切り捨てた結果を表す。  〔手順〕  (1) “Ｈ←[データ数÷３]“とする。  (2) データ列を，互いにＨ要素分だけ離れた要素の集まりから成る部分列とし，それぞれの部分列を，挿入法を用いて整列する。  (3) “Ｈ←[Ｈ÷３]”とする。  (4) Ｈが０であればデータ列の整列は完了し，０でなければ(2)に戻る。  ア　２ イ　３ ウ　４ エ　５  シェルソートは、挿入ソートを改良したものです。一定間隔（通常は要素数の１／２）離れた要素同士で挿入ソートを行った後、間隔を半分にしながら挿入ソートを繰り返します。  問題に与えられたデータ列を〔手順〕に従って実行すると、次のようになります。  (1) データ数は９個あるので、〔９÷３〕→ Ｈより、Ｈは３となる。  (2) 挿入法を用いて整列する。  (3) 〔３÷３〕→ Ｈより、Ｈは１となる。  (4) Ｈ＝０でないため(2)に戻る。  (2) 挿入法を用いて整列する。  (3) 〔１÷３〕→ Ｈより、Ｈは０となる。  (4) Ｈ＝０であるため、データ列の整列を完了する。  したがって、手順(3)は２回実行されます。  応用情報　平成31年度春　問6　[出題頻度：★☆☆]  解答－ア |

🏋プラスアルファ

**●マージソート**

マージソートは、要素全体を分割し、各部分内で整列しておき、その後、分割した部分を併合（マージ）することで整列する方法です。データの数をｎとすると計算量は*O*（ｎlog2ｎ）となります。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題　🏋プラスアルファ  データ列が整列の過程で図のように上から下に推移する整列方法はどれか。ここで，図中のデータ列中の縦の区切り線は，その左右でデータ列が分割されていることを示す。   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | ６ | １ | ７ | ３ | ４ | ８ | ２ | ５ | |  |  |  |  |  |  |  |  | | １　６ | | ３　７ | | ４　８ | | ２　５ | | |  |  |  |  |  |  |  |  | | １　３　６　７ | | | | ２　４　５　８ | | | | |  |  |  |  |  |  |  |  | | １　２　３　４　５　６　７　８ | | | | | | | |   ア　クイックソート イ　シェルソート  ウ　ヒープソート エ　マージソート  要素に分割されたデータの大小関係を見ながら併合（マージ）する整列方法を、マージソートと呼びます。  ア　クイックソートとは、データを適当な基準値により小さいグループと大きいグループに分割し、これを繰り返すことで整列をする方法です。  イ　シェルソートとは、一定間隔空けた要素で挿入ソートを行い、徐々にこの間隔を狭めていくことで整列をする方法です。  ウ　ヒープソートとは、２分木におけるヒープ木の構造を利用した整列方法で、未整列の要素でヒープを作成し、作成後のルートの要素を取り出しながら整列をする方法です。  ソフトウェア開発　平成18年度秋　問10　[出題頻度：★☆☆]  解答－エ |

### ２）探索（search）

配列やファイルなどのデータの集まりから、特定のデータを見つけるために走査する過程のことを探索といい、代表的な方法に次の３つがあります。

#### ①線形探索

線形探索は、**順次探索**（sequential search：シーケンシャルサーチ）とも呼ばれ、特定のデータを探索するために、配列の先頭（または末尾）から順番に調べていく方法です。配列要素がどのような並び順でも、探索することができます。

89

23

46

77

51

94

35

１

２

３

４

５

６

７

Ｔ

94

先頭から順番に比較

Ｘ

線形探索のイメージ

|  |
| --- |
| 例題  配列上に不規則に並んだ多数のデータの中から，特定のデータを探し出すのに適したアルゴリズムはどれか。  ア　２分探索法 イ　線形探索法  ウ　ハッシュ法 エ　モンテカルロ法  不規則に配列されている多数のデータの中から特定のデータを探索するのには線形探索法が最適です。線形探索法は、データの先頭から順に特定のデータを探索していくので配列が順番に並んでいる必要はありません。  ア　２分探索法は、あらかじめデータが整列されている必要があります。  ウ　ハッシュ法とは、ハッシュ関数を用いてキー値からデータを格納する位置を求める手法です。  エ　モンテカルロ法とは、乱数を用いて近似値を求める手法です。最近では、乱数を用いて行うコンピュータシミュレーションを呼ぶ場合もあります。  ソフトウェア開発　平成19年度春　問13　[出題頻度：★☆☆]  解答―イ |

別冊演習ドリル 》 1-136,137

#### ②２分探索

２分探索（binary search：バイナリサーチ）は、探索の対象となる配列が、あらかじめ昇順または降順に並んでいるときに用いられます。常に配列の探索範囲の中央のデータと比較し、比較したデータよりも前半部分または後半部分を切り捨て、探索する範囲を徐々に狭めていく方法です。

１

２

３

４

５

６

７

Ｔ

89

23

46

77

51

94

35

77

Ｘ

Ｌ

Ｈ

Ｍ

探索範囲

①　探索範囲の上限、下限を示す変数をそれぞれＬ、Ｈとし、中央の値Ｍを求める。

最初はＬ＝１、Ｈ＝７、Ｍ＝(Ｌ＋Ｈ)÷２＝４（端数が出る場合は、小数点以下切り捨て）である。

23

46

51

35

探索範囲

１

２

３

４

５

６

７

Ｔ

89

77

94

77

Ｘ

Ｌ

Ｈ

Ｍ

切捨て

②　配列Ｔ(４)とＸを比較する。51＜77なので、Ｌを（Ｌ＝Ｍ＋１）移動して、前半部分を切り捨てる。

③　残った範囲で、ＬとＨからＭを求める。

探索範囲

１

２

３

４

５

６

７

Ｔ

89

23

46

77

51

94

35

77

Ｘ

ＬＭＨ

切捨て

④　配列Ｔ(６)とＸを比較する。89＞77なので、Ｈを（Ｈ＝Ｍ－１）移動して、後半部分を切り捨てる。

⑤　残った範囲でＬとＨからＭを求める。

⑥　以下、Ｔ(Ｍ)＝Ｘか、Ｌ＞Ｈになるまで、④と⑤を繰り返す。

２分探索のイメージ

|  |  |  |
| --- | --- | --- |
|  | 線形探索 | ２分探索 |
| 長　所 | 配列要素がどのような並び順でも探索が可能である。 | 探索が速い（配列の要素数が増えても比較回数はあまり変わらない）。 |
| 短　所 | 探索が遅い（配列の要素数が増えるとそれに比例して比較回数も増える）。 | 配列要素が昇順または降順になっていなければ探索ができない。 |
| 平均比較回数 | ｎ÷２回 | [log２ｎ]回 |
| 最大比較回数 | ｎ回 | [log２ｎ]＋１回 |
|  |  | ※[　　]は小数点以下切捨てを意味する。 |

線形探索と２分探索の特徴

|  |
| --- |
| 例題  あらかじめ整列された1000人の電話番号を２分探索法を用いて探索するとき，最大何回の比較で見つかるか。  ア　7 イ　8 ウ　9 エ　10  整列済みデータを２分探索法を用いて検索した場合、平均比較回数は[log2ｎ]回、最大比較回数は[log2ｎ]＋１回です。  ｎ＝1000より  [log2ｎ]＋１＝[log21000]＋１  ２９＜1000＜２10より  log2２９＜log21000＜log2２10  ９＜log21000＜10  [log21000]＝９  [log21000]＋１＝10  第一種　平成11年度春　問6　[出題頻度：★★★]  解答－エ |

別冊演習ドリル 》 1-138～143

#### ③ハッシュ法

ハッシュ法は、データやキーの値から直接、格納位置（添字）を計算する方法であり、データの探索を高速に行うことができます。

この方法では、与えられたキーに特別な関数を用いて、格納位置を求めます。このときに用いる関数を**ハッシュ関数**といい、ハッシュ関数が返す値を**ハッシュ値**といいます。また、ハッシュ関数を用いてデータの格納位置を求めることを**ハッシング**といいます。なお、キーの分布によらず、できるだけ均等でかつ広範囲なハッシュ値を与えるハッシュ関数が“良い”ハッシュ関数といえます。

ハッシュ関数

キー

データ

キー値

ハッシュ値

ハッシュ値で

直接アクセス

ハッシュ表

キー

データ

ハッシュ値が示す位置

ハッシュ法の概念

ハッシュ法は、データを格納する際、ハッシュ関数によって特定の範囲の値にキーを変換するため、異なるキーにもかかわらず同一のハッシュ値が得られることがあります。これを**衝突**（**コリジョン**:collision）といい、先に格納されていたデータを**ホームレコード**、後から変換されたデータを**シノニムレコード**といいます。

ハッシュ法による探索に要する平均比較回数はシノニムレコードが発生する確率に左右されます。衝突がまったくない場合には１回の比較で探索できますが、衝突が多くなるとそれだけ比較回数も増加します。

シノニムレコードが発生する確率を最小にするには、ハッシュ値に偏りがなく、一様に分布している必要があります。

次の手順でハッシュ表にデータを格納する場合

ハッシュ関数：ハッシュ値＝（キー値÷10）の余り

ハッシュ表

０

１

２

３

３

21

12

⋮

①キー値＝12の場合

12÷10＝１余り２

ハッシュ表の２番目に格納→ホームレコード

②キー値＝21の場合

21÷10＝２余り１

ハッシュ表の１番目に格納→ホームレコード

③キー値＝42の場合

42÷10＝４余り２

ハッシュ表の２番目に格納→シノニムレコード

衝突

衝突（例）

🏋プラスアルファ

**●衝突時の回避方法**

衝突が発生した場合、何らかの方法でこれを回避しなければなりません。この方法には、チェイン法やオープンアドレス法があります。

**ⅰ）チェイン法**

チェイン法では、同じハッシュ値をもつデータは、ポインタによってリストにつながれており、ハッシュ表にはリストの最初  
　　のデータを示すポインタだけを入れておきます。

×

⋮

データＡ

データＢ

データＣ

データＤ

データＥ

データＦ

ハッシュ表

１

２

３

ｎ

**ⅱ）オープンアドレス法**

オープンアドレス法では、衝突が発生した場合、再ハッシュを行い、ハッシュ表の空き領域を探し、格納します。

①データａの

ハッシュ値＝１

②データｂの

ハッシュ値＝３

③データｃの

ハッシュ値＝１

衝突

ハッシュ表

１

２

３

４

５

６

ｎ

ａ

ｂ

ｃ

：

再ハッシュ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  10進法で５桁の数a1 a2 a3 a4 a5を，ハッシュ法を用いて配列に格納したい。ハッシュ関数をmod(a1＋a2＋a3＋a4＋a5，13)とし，求めたハッシュ値に対応する位置の配列要素に格納する場合，54321は配列のどの位置に入るか。ここで，mod(*x*，13)の値は，*x*を13で割った余りとする。   |  |  | | --- | --- | | 位置 | 配列 | | 0 |  | | 1 |  | | 2 |  | | ･･･ | ･･･ | | 11 |  | | 12 |  |   ア　１ イ　２ ウ　７ エ　11  ５桁の数a1 a2 a3 a4 a5のハッシュ関数は、次の式で定義されています。  ハッシュ関数＝mod(ａ１＋ａ２＋ａ３＋ａ４＋ａ５，13)  上記の式に“54321”を当てはめると、  mod(5＋4＋3＋2＋1，13)＝mod(15，13)＝２  となり、解答はイとなります。  基本情報　令和4年度サンプル　問7　[出題頻度：★★★]  解答－イ |

別冊演習ドリル 》 1-144～150

🏋プラスアルファ

**●グラフのアルゴリズム**

グラフとは、点（節点）とその間の辺（枝）の集まりから構成される図形をいいます。

全ての節点や枝を１回だけ通るように、グラフを調べることを、グラフの探索といいます。グラフの探索アルゴリズムの代表的なものに、深さ優先探索と幅優先探索があります。

**①深さ優先探索**

深さ優先探索は、まず子の１つを調べ、次にその子のさらに子を調べるというように、下に向かって調べて行き、行き止まったら戻っ  
　　て他の子を探索する方法です。

１

４

５

２

６

７

３

深さ優先探索（例）

深さ優先探索で各節を訪問した場合、訪問した節を出力する順序により、次のような方法があります。

**ⅰ）先行順**

先行順は、深さ優先探索で各節を訪問した際に「親節の値→左部分木の各節の値→右部分木の各節の値」の順に出力しま  
　　す。そのため、上記の二分木の例の場合、先行順での出力順序は「１→２→４→５→３→６→７」となります。

**ⅱ）中間順**

中間順は、深さ優先探索で各節を訪問した際に、「左部分木の各節の値→親節の値→右部分木の各節の値」の順に出力しま  
　　す。そのため、先の二分木の例の場合、中間順での出力順序は「４→２→５→１→６→３→７」となります。

**ⅲ）後行順**

後行順は、深さ優先探索で各節を訪問した際に「左部分木の各節の値→右部分木の各節の値→親節の値」の順に出力しま  
　　す。そのため、先の二分木の例の場合、後行順での出力順序は「４→５→２→６→７→３→１」となります。

**②幅優先探索**

幅優先探索は、まず自分の子を全て探索してから、子の子を全て探索し、さらに子の子の子を全て探索するというように、同じ深  
　　さのものを全て探索してから次の階層を探索する方法です。

１

４

５

２

６

７

３

幅優先探索（例）

なお、幅優先探索は、探索する深さが大きくなると必要となる記憶領域が指数的に増加するため、木の高さが小さく、１つの節  
　　から伸びる枝の数が少ない場合でないと実用的ではありません。

また、深さ優先探索が平均で木の半分を探索するのに対して、幅優先探索は木の末端の手前までの全部と末端の半分を探索  
　　するため、幅優先探索の方が深さ優先探索に比べ、一般的に必要となる探索時間も多くなります。

|  |
| --- |
| 例題　🏋プラスアルファ  図の２分木を深さ優先の先行順で探索を行ったときの探索順はどれか。ここで，図中の数字はノードの番号を表す。  4  5  2  1  3  6  ア　1，2，3，4，5，6 イ　1，2，4，5，3，6  ウ　4，2，5，1，3，6 エ　4，5，2，6，3，1  木の巡回順は幅優先順と深さ優先順に分けられます。先行順は、深さ優先順の１つであり、親、左部分木、右部分木の順に巡回します。  *４*  *６*  *５*  *１*  *２*  *３*  ４  ５  ２  １  ３  ６  節の左側にあるアンダーラインの数字は、巡回する番号を表します。  ア　幅優先順：同じ深さの節を順に巡回します。  ウ　深さ優先順（中間順）：左部分木、親、右部分木の順に巡回します。  エ　深さ優先順（後行順）：左部分木、右部分木、親の順に巡回します。  ソフトウェア開発　平成19年度秋　問13　[出題頻度：★☆☆]  解答－イ |

別冊演習ドリル 》 1-151

🏋プラスアルファ

**●文字列検索**

文字列検索とは、与えられた文字列の中から、検索対象の文字列を探す方法です。

**①ブルートフォースアルゴリズム**（brute-force algorithm）

最も単純な文字列検索の方法は、与えられた文字列の先頭と検索対象の文字列の先頭を重ね合わせて１文字ずつ照合し、  
　　対象文字列の末尾まで一致すれば検索成功、途中で文字が違えば検索失敗と判断して開始位置を１文字ずらして再度比較  
　　し、対象文字列が見つかるまで続ける方法です。検索対象の文字列がｍ文字、与えられた文字列がｎ文字の場合、処理量が

　　最大になるのはｍ文字で検索失敗となることが（ｎ－ｍ＋１）回繰り返されるため、計算量はＯ（ｍｎ）となります。

a

c

d

b

a

b

a

c

a

b

c

a

c

b

d

a

b

a

c

a

b

c

a

c

b

d

a

b

a

c

a

b

c

b

a

c

a

c

d

b

a

b

a

c

(1)失敗

(2)失敗

(3)失敗

(4)成功

**②BM**（ボイヤ・ムーア）**法**

BM法も、与えられた文字列の先頭と検索対象の文字列の先頭を重ね合わせることは同じですが、検索対象の文字列の末尾か  
　　ら先頭に向かって照合し、先頭まで一致すれば検索成功、途中で文字が違えば検索失敗と判断し、不一致となった文字が検索  
　　対象の文字列に含まれていない場合、そこまでの範囲は一致するはずがないので、次の位置までずらして再度照合を始めます。  
　　この方法では最悪の場合、計算量はＯ（ｎ）となります。

a

c

d

b

a

b

a

c

a

b

c

a

b

c

a

c

d

b

a

b

a

c

(1)失敗

(2)成功

|  |
| --- |
| 例題　🏋プラスアルファ  文字列Aが“aababx△”，文字列Bが“ab△”であるとき，流れ図の終了時点の*k*は幾らか。ここで，文字列の先頭の文字を１番目と数えるものとし，A[*i*]はAの*i*番目の文字を，B[*j*]はBの*j*番目の文字を，“△”は終端を示す文字を表す。  *i*－*j*＋2 → *i*  1 → *j*  1 → *i*  1 → *j*  0 → *k*  2 → *jmax*  開始  A[*i*]：B[*j*]  A[*i*]＝“△”  or B[*j*]＝“△”  B[*j*]＝“△”  *i*＋1 → *i*  *j*＋1 → *j*  *i*－*jmax* → *k*  終了  ≠  ­＝  Yes  Yes  No  No  ア　０ イ　１ ウ　２ エ　４  問題文のデータで流れ図をトレースすると次のようになります。  ①文字列Ａの１文字目における処理  ａ．ｉ＝１、ｊ＝１  Ａ[1]＝Ｂ[1]であるため、ｉ、ｊに１を加算  ｂ．ｉ＝２、ｊ＝２  Ａ[2]≠Ｂ[2]であるため、（ｉ－ｊ＋２→ｉ）よりｉに２、（１→ｊ）よりｊに１を設定  ②文字列Ａの２文字目における処理  ａ．ｉ＝２、ｊ＝１  Ａ[2]＝Ｂ[1]であるため、ｉ、ｊに１を加算  ｂ．ｉ＝３、ｊ＝２  Ａ[3]＝Ｂ[2]であるため、ⅰ、jに１を加算  ｃ．ｉ＝４、ｊ＝３  Ｂ[3]＝“△”が成立するため、（ｉ－jmax→ｋ）よりｋに２を設定  なお、本問は、文字列Ａの中から文字列Ｂのパターンが含まれているかを検索し、一致する文字列があればｋにその文字列の文字列Ａの開始位置を、一致する文字列がなければｋに０を設定する流れ図です。  基本情報　平成19年度春　問13　[出題頻度：★★☆]  解答－ウ |

別冊演習ドリル 》 1-152,153

# 3. プログラミング

## 1. プログラム構造

学習のポイント

✅ プログラムの性質を覚えよう！

✅ 日本語名称とカタカナ名称（例えば「再使用可能」と「リユーザブル」）をセットで覚えよう！

コンピュータ上で実行されるプログラムには、いくつかの性質があります。実行するプログラムの性質により仕様が変わってくるので、注意が必要です。

### １）再使用可能（リユーザブル）プログラム

再使用可能（リユーザブル）プログラムは、プログラムを実行するごとに主記憶にロードせずに、一度ロードした後は複数のプログラム（プロセス）で共通に使用でき、何度でも繰り返し使用することができるプログラムです。再使用可能プログラムには、次のような種類があります。

#### ①逐次再使用可能プログラム

逐次再使用可能プログラムは、１つのプログラムを同時に１つのプロセスだけが使用できるプログラムです。逐次再使用可能プログラムでは、変数の初期化などをプログラムの開始時に行い、処理の整合性を保っています。

使用後

プログラム

Ａ＋２→Ａ

Ｂ＋３→Ｂ

プロセスＸ

Ａ

Ｂ

０

２

０

３

プログラム

Ａ＋２→Ａ

Ｂ＋３→Ｂ

プロセスＹ

初期化される

Ａ

Ｂ

０

０

逐次再使用可能プログラム（例）

#### ②再入可能（リエントラント）プログラム

再入可能（リエントラント）プログラムは、１つのプログラムを同時に複数のプロセスが使用できるプログラムです。再入可能プログラムでは、プログラムを変数部分と命令部分に分割し、プロセスごとに変数部分を割り当て、命令部分のみを共有することで同時処理を実現しています。

プログラム

Ａ＋２→Ａ

Ｂ＋３→Ｂ

プロセスＸ

プロセスＹ

Ａ

Ｂ

０

２

０

３

Ａ

Ｂ

０

２

０

３

プロセスごとに変数を用意する

同時に複数のプロセスを投入する

再入可能プログラム（例）

### ２）再帰的（リカーシブ）プログラム

**再帰**（さいき）は、プログラムの中から自分自身を呼び出すことです。このようなアルゴリズムを再帰的（recursive：**リカーシブ**）なアルゴリズムといい、再帰的に呼び出すことを**再帰呼出し**（recursive call）といいます。

サブルーチンａ

サブルーチンａ

の実行

主プログラム

サブルーチンａ

の実行

（戻り）

サブルーチンａ

サブルーチンａ

の実行

サブルーチンａ

サブルーチンａ

の実行

（再帰）

（戻り）

（再帰）

（戻り）

再帰呼出しのイメージ

なお、再帰的な処理を実現するためには、実行途中の状態をLIFO（後入れ先出し）方式で記憶領域に保存しておく必要があります。

|  |
| --- |
| 例題  自然数ｎに対して，次のとおり再帰的に定義される関数f（ｎ）を考える。f（5）の値はどれか。  f（n）：if　n≦１　then return　１　else　return　n＋f（ｎ－１）  ア　６ イ　９ ウ　15 エ　25  問題文の式を、再帰的に順を追って計算します。  ①　f（５）＝５＋f（４） 　　　　　　　　＝５＋10＝15  ⑥  ⑦  ⑦  ⑧  ②　f（４）＝４＋f（３） 　　　　　　　　＝４＋６＝10  ③　f（３）＝３＋f（２） 　　　　　　　＝３＋３＝６  ④　f（２）＝２＋f（１）＝２＋１＝３  ⑤　f（１）＝１  基本情報　令和4年度サンプル　問8　[出題頻度：★★☆]  解答－ウ |

別冊演習ドリル 》 1-154～160

再帰的（リカーシブ）プログラムは、プログラム実行中に、自分自身を呼び出す（実行する）ことができるプログラムです。同じ変数名に違う値が格納されるため、呼び出すごとに同じ変数名であるが異なった領域を確保しなければなりません。そのため、スタックを使用します。これを使用することによって、効率的なプログラムを記述することができます。

プログラムＸ

プログラムＸ

条件

出口

Yes

No

再帰的プログラム（例）

### ３）再配置可能（リロケータブル）プログラム

再配置可能（リロケータブル）プログラムは、主記憶上のどの位置に格納されても、正しく実行されるプログラムです。

|  |
| --- |
| 例題  あるプログラムＡの処理が終了していないときに，別のプログラムから再度呼び出されても正しく動作するとき，このプログラムＡの性質を何と呼ぶか。  ア　再帰的 イ　再使用可能 ウ　再入可能 エ　再配置可能  再入可能は、複数のプログラムが同時に使用できるような、あるいは繰り返し実行できるような構造を表し、この構造をもったプログラムを再入可能プログラムといいます。  ア　再帰的は、あるモジュール（関数または手続き）の実行中に自分自身を呼び出したり、あるいは呼び出したモジュールからさらに呼び出されたりするような構造であり、この構造をもったプログラムを再帰的プログラムといいます。  イ　再使用可能は、一度実行した後、ロードし直さずに再び実行を繰り返しても正しい結果が得られる構造であり、この構造をもったプログラムを再使用可能プログラムといいます。  エ　再配置可能は、任意の記憶領域に割り付けられているプログラムの格納位置を、別の位置に再設定する再配置ができる構造であり、記憶領域のどの領域に格納されていても実行可能なプログラムを、再配置可能プログラムといいます。  基本情報　平成22年度春　問8　[出題頻度：★★☆]  解答－ウ |

別冊演習ドリル 》 1-161～164

# 4. プログラム言語

ソフトウェア開発は、プログラム言語を使って行います。プログラム言語は、コンピュータに行わせる処理の流れ（アルゴリズム）を、命令（コマンド）の組合せで記述するためのものです。

## 1. プログラム言語の種類と特徴

学習のポイント

✅ Javaを中心に用語を覚えよう！

プログラム言語の体系を示すと、次のようになります。

プログラム言語

汎用プログラム言語

高水準言語

低水準言語

機械語

アセンブラ言語

手続き型言語

RPG

APL

LISP

Prolog

C++

Java

等々・・・

Fortran

COBOL

Pascal

C

PL/Ⅰ

ALGOL

BASIC

等々・・・

非手続き型言語

スクリプト言語

第４世代言語

マークアップ言語

SGML

HTML

XML

Perl

JavaScript　等々・・・

プログラム言語の体系

### １）低水準言語

低水準言語は、コンピュータに強く依存した、特定のハードウェア固有の言語です。通常、０と１の２進数や16進数、あるいはコードなどで記述するため、人間にとって判断しにくい言語です。

### ２）高水準言語

高水準言語は、低水準言語と異なり、特定のハードウェアを意識することなく、目的とする処理を容易に記述できるように設計された言語です。問題解決に専念できることから、問題向き言語とも呼ばれます。

#### ①手続き型言語

手続き型言語は、問題を解決するための手順を、命令を並べて記述することによってプログラムを作成する言語です。現在、使用されているプログラムの多くが、Fortran、COBOL、BASIC、Cなどの手続き型プログラム言語で記述されています。

#### ②非手続き型言語

非手続き型言語は、処理手順を記述するのではなく、機能を中心に記述することでプログラムを作成する言語です。その中でも、現在主役をなす言語が、オブジェクト指向型プログラム言語である**Java**です。オブジェクト指向型プログラム言語は、プログラムが扱う対象をオブジェクト（データとそれに対する処理をまとまたもの）としてとらえ、オブジェクトの組合せでプログラムを作成します。

Javaで書かれたプログラム（**Javaアプリケーション**）は、コンパイラ（翻訳プログラム）によって開発環境や実行環境に依存しない中間コードであるバイトコードに変換され、**仮想マシン**（**VM**：Virtual Machine）と呼ばれるインタプリタ（解釈プログラム）によって解釈（ネイティブコードに変換）、実行されます。そのため、実行環境（OS）ごとに仮想マシンを用意すれば、機種に関係なく実行できます。さらに、仮想マシンがメモリ領域の開放を**ガーベジコレクション**という機能で担当するため、プログラマはメモリ領域の大きさを意識せずにプログラムの作成を行えます。また、セキュリティ機能や豊富なネットワーク関連の機能が標準で搭載されており、インターネットや分散処理システムなどのネットワーク環境で利用されることを意識した仕様になっています。

Webサーバからダウンロードされ、ブラウザ上で動作するJavaプログラムを**Javaアプレット**と呼びます。なお、アプレットとは、サーバからダウンロードされてクライアント上で実行される小さいアプリケーションプログラムを意味します。また、Webサーバ上で実行できるモジュール化されたJavaプログラムを**Javaサーブレット**と呼びます。Javaサーブレットは、一度呼び出されると、そのままメモリに常駐するため、高速な処理が可能で、複数のユーザ間で情報を共有することもできます。

Javaのプログラムにおいて、よく使われる機能などを部品化し、再利用できるようにコンポーネント化（クラス化）するための仕様を**JavaBeans**（ジャバビーンズ）といい、開発されたプログラムを、アプリケーションの部品として取り扱うための規約を、**EJB**(Enterprise JavaBeans)といいます。

|  |  |
| --- | --- |
| Javaアプリケーション | Javaで書かれたプログラム、独立した単体のプログラムとして実行 |
| 仮想マシン（VM） | Javaのインタプリタ |
| Javaアプレット | Webサーバからダウンロードされ、ブラウザ側で実行するJavaプログラム |
| Javaサーブレット | サーバ上で実行されるJavaプログラム |
| Java Beans | Javaのプログラムにおいて、よく使う機能などを部品化し、再利用できるようにコンポーネント化する仕様 |

Javaに関する用語のまとめ

|  |
| --- |
| 例題  Javaの特徴はどれか。  ア　オブジェクト指向言語であり，複数のスーパクラスを指定する多重継承が可能である。  イ　整数や文字は常にクラスとして扱われる。  ウ　ポインタ型があるので，メモリ上のアドレスを直接参照できる。  エ　メモリ管理のためのガーベジコレクションの機能がある。  ア　言語構造が複雑になりすぎるため、多重継承はできません。  イ　基本データ型をクラスとすることはできません。  ウ　直接アドレスを参照するようなポインタの概念は存在しません。  基本情報　平成30年度秋　問8　[出題頻度：★★★]  解答－エ |

別冊演習ドリル 》 1-165～169

### ３）スクリプト言語

スクリプト言語は、機械語への変換作業を省略して簡単に実行できるようにした、簡易プログラムを記述するための言語です。小規模なプログラムをすばやく作成することがおもな目的となっている言語で、利用者がマウスやキーボードで入力を行うと、即座に処理を行うイベント駆動型の言語です。代表的なものに、PerlやJavaScriptがあります。

**JavaScript**（ジャバスクリプト）を用いると、HTMLファイル内に直接プログラムを記述し、ブラウザで実行することが可能です。そのため、HTMLだけでは不可能なブラウザ側での入力データの検査もJavaScriptを用いることで可能です。また、JavaScriptとWeb関連のXMLなどの技術を組み合わせた**Ajax**（Asynchronous JavaScript＋XML：エイジャックス、アジャックス）を用いると、画面遷移を伴わない動的なユーザインタフェースを実現することもできます。

|  |
| --- |
| 例題  JavaScriptの非同期通信の機能を使うことによって，動的なユーザインタフェースを画面遷移を伴わずに実現する技術はどれか。  ア　Ajax イ　CSS ウ　RSS エ　SNS  イ　CSS（Cascading Style Sheets）は、Webページのレイアウトを定義するもので、フォントや文字の大きさなど文書の見栄えに関する情報を、HTML文書本体から切り離したものです。  ウ　RSS（Rich Site Summary）は、Webサイトの見出しや要約などを簡略化して配信するためのXMLベースのフォーマットです。  エ　SNS（Social Network Service）は、コミュニケーションを円滑にする手段や場所を提供するコミュニティ型のWebサイトのことです。  基本情報　平成22年度秋　問49　[出題頻度：★★☆]  解答－ア |

別冊演習ドリル 》 1-170,171

## 2. プログラム言語の制御構造

学習のポイント

✅ 用語を覚え、例題のような問題が解けるように準備しよう！

メインプログラム（メインルーチン）からサブプログラム（サブルーチン、関数）を呼び出す方法には、値呼出しと参照呼出しがあります。

**値呼出し**では、サブプログラムを呼び出すときに、引数（データ）のコピーを渡して、引数の記憶場所（アドレス）は渡しません。その結果、サブプログラム内に独立した新しい変数が作成されます。そのため、サブプログラム中の変数の値を変更しても、メインプログラムの変数が変化することはありません。

これに対し**参照呼出し**では、サブプログラムを呼び出すときに、引数のアドレスが渡されます。そのため、メインプログラムとサブプログラムの間で、変数の記憶域が共有され、サブプログラムでその変数に対し行った操作は、メインプログラムでもそのまま反映されます。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 例題  メインプログラムを実行した後，メインプログラムの変数X，Yの値は幾つになるか。ここで，仮引数Xは値呼出し（call by value），仮引数Yは参照呼出し（call by reference）であるとする。  メインプログラム 手続add（X，Y）  *X＝X＋Y；*  *Y＝X＋Y；*  *return；*  *X＝2；*  *Y＝2；*  *add(X，Y)；*   |  |  |  | | --- | --- | --- | |  | X | Y | | ア | 2 | 4 | | イ | 2 | 6 | | ウ | 4 | 2 | | エ | 4 | 6 |   値呼出しは変数の内容を受け渡す呼出し方法で、値は呼び出す方から呼び出される方への一方通行となります。それに対して、参照呼出しは変数のアドレスを受け渡す呼出し方法で、呼び出された方で行った操作は、呼び出す方にも反映されます。  関数addにおいて  X＝X＋Y＝2＋2＝4  Y＝X＋Y＝2＋4＝6  となりますが、Xに対する変更は値呼出しであるためメインプログラムのXには影響しません。一方、Yに対する変更は参照呼出しであるためメインプログラムのYにも影響を与えます。したがって、メインプログラムに戻った後の変数X、Yの値は、それぞれ２、６となります。  応用情報　平成25年度春　問20　[出題頻度：★☆☆]  解答―イ |

別冊演習ドリル 》 1-172,173

# 5. その他の言語

## 1. マークアップ言語

学習のポイント

✅ XMLを中心に、それぞれの特徴を覚えよう！

マークアップ言語は、文書中のある部分に関連する情報を文書自体に埋め込むために使われる言語です。指定部分の前後にタグという記号を挿入し、データの種類やレイアウト情報などの属性を記述します。

### １）HTML（Hyper Text Markup Language）

HTMLは、Webページを記述するための言語で、他のページや情報を自動的に呼び出すことができます。

### ２）SGML（Standard Generalized Markup Language）

SGMLは、文書の構造や意味を定義するマークアップ言語で、1986年にISO（国際標準化機構）によって国際規格が制定されました。文書中のデータをコンピュータ間で交換しやすいように仕様が定められています。複数の文書からタイトルの一覧を作ることや、長い文書の中の重要箇所だけを検索する、といったことができます。

|  |
| --- |
| 例題  タグを使って文書の論理構造や属性を記述する方法を定めた国際規格であって，電子的な文書の管理や交換を容易に行うための文書記述言語はどれか。  ア　DML イ　HTML ウ　SGML エ　UML  ア　DML（Data Manipulation Language）は、データ操作言語の略称で、DBMS（DataBase Management System）のもとで、データベースのデータを読み書きするための言語です。  イ　HTMLは、ブラウザによって閲覧できるハイパテキスト形式の文書を記述するための言語です。  エ　UML（Unified Modeling Language）は、オブジェクト指向によるソフトウェア開発において、プログラムの構造を統一された図で表すモデリング技法のことです。  基本情報　平成20年度春　問39　[出題頻度：★☆☆]  解答－ウ |

別冊演習ドリル 》 1-174

### ３）XML（eXtensible Markup Language）

XMLは、HTMLと同様に、SGMLをもとに作成されたマークアップ言語ですが、HTMLと異なり、ユーザが独自のタグを使い、データの意味や構造を自由に定義できます。

インターネット環境との親和性が高く、Webデータや電子商取引のデータなど、さまざまな文書を記述することができます。

XML文書の最小単位は要素であり、要素は、タグをつけたい文字データの前に開始タグと呼ばれる<要素名>をつけ、終わりに終了タグと呼ばれる</要素名>をつけます。また、開始タグと終了タグは必ず対になっていることが必要です。ただし、データをもたないタグだけの要素(空要素)もあり、その場合は、<要素名/>または、<要素名></要素名>のように記述します。なお、要素の中に子要素をもつこともできます。

形式上、正しい文法で記述されたXML文書を、整形式文書と呼びます。また、XML文書では、文書構造を定義するDTD（Document Type Definition）に適合している文書を「妥当な文書」と呼びます。これに対して、XML宣言のあと属性宣言を行い、扱う値の型などを定義していきますが、これを省略した場合には「妥当な文書ではない」といいます。

なお、HTMLやXMLなどWWWで使用される言語の仕様は、W3C(World Wide Web Consortium)により定められています。

|  |
| --- |
| 例題  XMLに関する記述のうち，適切なものはどれか。  ア　HTMLを基にして，その機能を拡張したものである。  イ　XML文書を入力するためには専用のエディタが必要である。  ウ　文書の論理構造と表示スタイルを統合したものである。  エ　利用者独自のタグを使って，文書の属性情報や論理構造を定義することができる。  XMLは、タグと呼ばれる文書構造や文字修飾を表現できる特別な文字列を利用者独自に作成し、文書の属性情報や論理構造を定義することができます。タグを用いる同様の言語に、HTMLがありますが、HTMLではタグの役割が初めから規定されているのに対し、XMLは、まずタグの役割を作成することから始める点が異なります。  基本情報　平成26年度秋　問8　[出題頻度：★★☆]  解答－エ |

別冊演習ドリル 》 1-175～179

### ４）スタイルシート

スタイルシートは、HTMLやXMLの要素をどのように表示するかを指示する場合に用いられます。W3Cが提唱するスタイルシートの仕様に、**CSS**（Cascading Style Sheets）があります。CSSは、Webページの見栄えを良くするための、文字の大きさ、文字の色、行間などに関する標準仕様です。

|  |
| --- |
| 例題  HTML文書の文字の大きさ，文字の色，行間などの視覚表現の情報を扱う標準仕様はどれか。  ア　CMS イ　CSS ウ　RSS エ　Wiki  CSS（Cascading Style Sheets）は、HTML文書本体からフォントや文字の大きさなどのWebコンテンツのレイアウトを定義する情報を切り離したものです。  ア　CMS（Contents Management System）は、Webコンテンツを構成するテキストや画像、レイアウト情報などを一元的に管理するシステムです。  ウ　RSS（Rich Site Summary）は、Webサイトの見出しや要約などを簡略化して配信するためのXMLベースのフォーマットです。  エ　Wikiは、複数の利用者がWebブラウザを用いて、コンテンツを追加したり、編集・削除したりできるWebコンテンツの管理システムです。  基本情報　平成28年度春　問24　[出題頻度：★★☆]  解答－イ |

別冊演習ドリル 》 1-180,181