第1章

基礎理論

ITについて学んだりIT関連の仕事をする際に知っておくと役に立つ分野，それが基礎理論です。直接目にする機会は少ないですが，基礎の理論ですので，いろいろな技術の仕組みや，本質的なことを知るときには役に立ちます。

分野は「基礎理論」と「アルゴリズムとプログラミング」の二つです。基礎理論では，離散数学や応用数学などの数学や，情報・通信・計測制御に関する理論について学びます。アルゴリズムとプログラミングでは，定番のデータ構造やアルゴリズム，プログラム言語やその他の言語でのプログラミングについて学びます。

1-1 基礎理論

IT全般を理解する上で基礎となる理論です。基礎理論を理解していると，コンピュータやシステムの仕組みが分かるだけでなく，ネットワークやデータベース，セキュリティなどの応用技術の学習にも役立ちます。

1-1-1 離散数学

勉強のコツ

最初は数学的な話が続くので，苦手な人は苦しいと思います。
この分野を理解できないと次に進めないというわけではないので，難しいと感じたら，「そのうち分かればいい」と，気楽に構えていきましょう。

離散数学とは，離散している（とびとびの）数字を使う数学です。コンピュータは，データを１か０，あるかないかの２種類の世界（ディジタル）でしか表現できません。そのような制限があるコンピュータで現実の連続した世界（アナログ）を表すために，いろいろな工夫が行われています。

例えば，数値は普段，人間の生活では10進数で考えられていますが，これをコンピュータで表現するために２進数に変換します。文字は文字コード，色は色コードに変換して表します。音声は，サンプリングという手法で特定の時間ごとに切り出して，ディジタルデータに変換します。画像や動画なども，ドットという単位に分解してディジタルデータに変換します。

コンピュータでデータを扱う場合は，変換をいろいろと行うため元のデータを完全に再現できないことも多くありますが，実用上は支障なく，十分役に立っています。

基数変換

勉強のコツ

基本的な基数変換は，基本情報技術者試験の勉強でも行います。応用情報技術者試験の勉強では，さらにそれを発展させていきます。基数変換を難しく感じる場合は，基本情報技術者試験の勉強を軽く復習してから学習することをおすすめします。

基数とは，数を表現するときに，けた上がりの基準になる数です。例えば，10進数では基数が10なので，0，1，2，3，4，5，6，7，8，9ときて10になるところでけた上がりをして10になります。同じように，2進数では基数が2なので，0，1ときて2になるところでけた上がりをして10になります。

例えば，2進数の$(110.01)_2$は，10進数では次のように表現できます。なお，下付きの数字は基数を表します。

$$(110.01)_2 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$
$$= (6.25)_{10}$$

負の数の表現（補数表現）

コンピュータ上で負の数を表す場合は，**補数**という考え方を使います。補数とは，ある自然数に足したときにけたが一つ上がる最も小さい数です。例えば，8けたの2進数$(00001101)_2$の2の補数は，足すことでけたが上がって9けたの最小値$(100000000)_2$になる値です。計算すると次のようになります。

	00001101	元の数	$(13)_{10}$
＋	11110011	2の補数	$(-13)_{10}$
	100000000	けたが一つ上がる最小値	

この補数を用いることによって，引き算を足し算で表現できます。例えば，$(25)_{10} - (13)_{10}$を計算するときには，次のように変換します。

$$(25)_{10} - (13)_{10} = (25)_{10} + (-13)_{10}$$

8けたの2進数に変換すると，先ほどの2の補数表現より$(11110011)_2 = (-13)_{10}$なので，次のようになります。

	00011001	引かれる数	$(25)_{10}$
＋	11110011	2の補数	$(-13)_{10}$
	100001100	計算結果	$(12)_{10}$

あふれたけた（9けた目）は無視される

引き算を足し算に直すことで計算の種類を減らすことができ，計算に必要なコンピュータの回路を単純にできます。

なお，2の補数の求め方としては，各ビットを反転して，それに1を加えることによって，簡単に計算することができます。

小数の表現

小数をコンピュータ内部で表現する方法には，次の2種類があります。

- 固定小数点数表現
- **浮動小数点数表現**

参考

補数の考え方は，実は小学校のときに習っています。10進数では，1の補数は9，2の補数は8，3の補数は7……というかたちで覚えて，引き算で使ったと思います。これを2進数に応用したのが，2の補数です。

過去問題をチェック

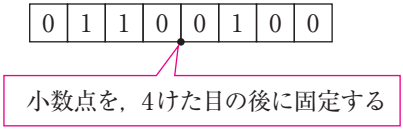
2の補数を使用する理由について，応用情報技術者試験 平成21年秋 午前 問1で問われています。

発展

コンピュータで数を表現するときは，あらかじめ「けた数」を決めておく必要があります。そして，決めたけた数からあふれたデータは，格納場所がないのでなかったものとして扱われます。

関連
プログラミング言語では，型の宣言によって，どちらの小数点数表現が使われるかが決まります。
C言語やC++，Java言語で用いられるfloat型，double型は，浮動小数点を表すデータ型です。
COBOLなど事務処理系の言語では，固定小数点数表現が使われることが多いです。

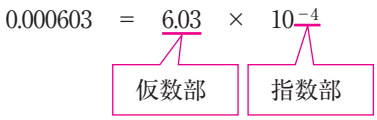
固定小数点数表現とは，あらかじめ小数点の位置を決めておき，その位置に合わせてデータを表現する方法です。例えば，2進数の$(110.01)_2$を表現するとき，8けた分の場所を確保し，4けた目の後を小数点にすると決めると，次のようになります。



固定小数点数表現

小数点の位置が決まっているためデータの解釈は簡単なのですが，その分，表現できる数値の範囲が限られてしまいます。例えば，上図の8けたで数値を表現すると，最大値でも$(1111.1111)_2 = (15.9375)_{10}$となり，大きな数は表現できません。

それに対し浮動小数点数表現では，指数部と仮数部を使って数値を表現します。例えば，10進数の0.000603は，10進数の浮動小数点数表現を使うと次のように表されます。



浮動小数点数表現

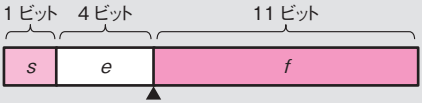
2進数の場合には，指数部を2のべき乗のかたちにし，符号部（数値がプラスなら0，マイナスなら1）と合わせて表現します。

具体的な方法はいろいろありますが，決められた形式のルールに従って数値を変換します。正しい形式に変換するこの作業を**正規化**と呼びます。

実際の問題を例に，正規化の作業を行ってみましょう。

問題

図に示す16ビットの浮動小数点形式において，10進数0.25を正規化した表現はどれか。ここで，正規化は仮数部の最上位けたが1になるように指数部と仮数部を調節する操作とする。



s：仮数部の符号（0：正，1：負）
e：指数部（2を基数とし，負数は2の補数で表現）
f：仮数部（符号なし2進数）

ア

0	0001	10000000000

イ

0	1001	10000000000

ウ

0	1111	10000000000

エ

1	0001	10000000000

（平成22年春 応用情報技術者試験 午前 問2）

解説

この問題の正規化では，仮数部の最上位けたが1になるように指数部と仮数部を調節します。つまり，10進数の0.25を2進数に直すと，$(0.25)_{10} = (0.01)_2$となるので，これを仮数部の最上位けたが1になるように浮動小数点数表現にすると，次のようになります。

$$(0.01)_2 = (0.1)_2 \times 2^{-1}$$

仮数部　指数部

また，指数部は$(-1)_{10}$なので，問題文中のe（指数部）の表記に「2を基数とし，負数は2の補数で表現」とあり，またeは4ビットで表されることから，$(1)_{10} = (0001)_2$を2の補数で表現します。

$$(-1)_{10} = (10000)_2 - (0001)_2 = (1111)_2$$

したがって，浮動小数点数表現で$(0.25)_{10}$を表すと，

s：仮数部の符号 …… 正の数なので「0」
e：指数部 …… 先ほど求めた2進数「1111」
f：仮数部 …… 1だが，後ろに0を補って「10000000000」

となり，合わせて「0 1111 10000000000」となります。

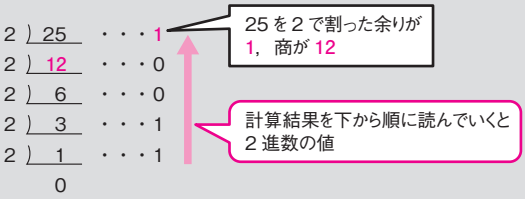
≪解答≫ ウ

コラム 基数変換の方法——10進数から2進数へ

2進数から10進数の基数変換は，P.20で解説したように，単純にけたごとに基数をかけ算してそれを足せば求めることができます。しかし，10進数から2進数に変換するときには少し工夫が必要です。基本情報技術者試験の範囲ではあるのですが，応用情報技術者試験でもよく出てくる内容なので，ここで改めて学習しておきましょう。

●整数の場合

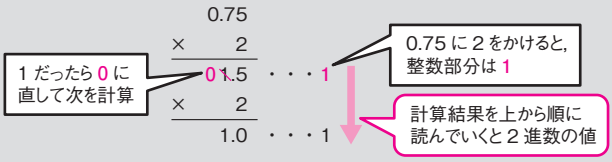
順に2で割って余りを求め，最後に商が0になったら終わりです。例えば，10進数の25を2進数にする場合は次のように計算を行います。



余りを下から順に読んでいった$(11001)_2$が，$(25)_{10}$の変換結果になります。

●小数の場合

小数の場合は，逆に2をかけていきます。かけ算をして1を超えるかどうかを求め，1を超えたらそれを0に直して順に計算を行い，小数部分が0になれば終わりです。例えば，10進数の0.75を2進数にする場合は次のように計算を行います。



整数部分を上から順に読んだ$(0.11)_2$が，$(0.75)_{10}$の変換結果になります。

このような計算方法を知っていると実際に計算するときに便利なので，覚えておきましょう。

勉強のコツ

基数変換などは基本情報技術者試験の範囲ですが，応用情報技術者試験でも出てきます。

基礎理論を難しく感じる場合は，基本情報技術者試験の勉強からやり直すのがベストです。ただ，応用情報技術者試験で問われることは基礎理論だけではありませんし，基礎理論をきちんと理解するには高校レベルの数学の知識が必要です。

割り切って，ここで紹介する計算方法だけでもマスターしておくと，いくつかの問題は解くことができます。完璧にする必要はないので，できる範囲で基礎理論の分野を学習していきましょう。

基数変換と誤差

コンピュータで数値を表現する際には，誤差に注意する必要があります。例えば小数の場合は，**10進数から2進数へ基数変換するだけでも誤差が出ることがあります**。

10進数の0.4を2進数に変換する場合を考えてみましょう。0.4という数は有限のけたで表されているので，こういった数のことを**有限小数**と呼びます。0.4を2進数に変換すると，割り切れず循環する，次のような数になります。

$$(0.4)_{10} = (0.01100110011001100110)_2 = (0.\dot{0}1\dot{1}0)_2$$

有限小数で表せない小数を**無限小数**といいますが，そのうち上記のように0110の部分が繰り返される小数のことを**循環小数**と呼びます。0.4という有限小数は，10進数を2進数に変換しただけで循環小数となるのです。

逆に2進数から10進数への基数変換の場合には，有限小数を変換すると必ず有限小数になります。

過去問題をチェック
基数変換における有限小数が無限小数になる変換については，ソフトウェア開発技術者試験 平成20年秋 午前 問1で問われています。

発展
コンピュータ内部では，10進小数から2進小数への基数変換が自動的に行われることがあります。例えば，C言語やJavaなどでfloat型やdouble型を使うと，データは2進数で格納されるので数値に少し誤差が出ます。そうすると予期しない計算ミスが発生することがあるので，注意が必要です。

数値演算と誤差

コンピュータでは限られたけた数でデータを表現するので，いろいろな誤差や，表現しきれないことが出てきます。具体的には，次のようなことで誤差が発生します。

①けた落ち

値がほぼ等しい二つの数値の差を求めたとき，有効数字のけた数である**有効けた数が減る**ことによって発生する誤差です。

	256.432	有効けた数6けた
−	256.431	有効けた数6けた
	0.001	有効けた数が**1けた**になってしまう！

過去問題をチェック
けた落ちによる誤差については，ソフトウェア開発技術者試験 平成20年秋 午前 問2で問われています。

②情報落ち

絶対値が非常に大きな数値と小さな数値の足し算や引き算を行ったとき，**小さい数値が計算結果に反映されない**ことによって発生する誤差です。

	256.432	有効けた数6けた
+	0.000011	非常に小さい数
	256.432~~011~~	有効けた数の関係で無視される

間違えやすい
けた落ちと情報落ちは混同されやすいので，正確に覚えておきましょう。有効けた数が減るのがけた落ち，小さい数値の情報が落ちるのが情報落ちです。

③丸め誤差

指定された有効けた数で演算結果を表すために，切り捨て，切り上げ，四捨五入などで下位のけたを削除することによって発生する誤差です。

④打ち切り誤差

無限級数で表される数値の計算処理を有限の回数で打ち切ったことによって発生する誤差です。

⑤オーバーフロー

演算結果が有限のけた内で表せる範囲を超えることによって，使用している記述方法では値が表現しきれなくなることです。

⑥アンダーフロー

浮動小数点数演算において，演算結果の指数部が小さくなりすぎ，使用している記述方法では値が表現しきれなくなることです。つまり，浮動小数点数表記では，0や0に近い数値は表現することができません。

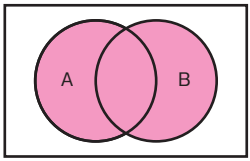
集合

集合は，ある条件で集まったグループです。例えば，集合A「コーヒーが好きな人」，集合B「紅茶が好きな人」とすると，コーヒーが好き，紅茶が好き，というのが条件です。

集合を視覚的に分かりやすく表すために，**ベン図**という図が用いられます。二つ以上の集合を組み合わせることが多く，代表的な組合せに次の五つがあります。

①和集合（A OR B，A＋B，A∪B，A∨B）

二つの集合を足したものです。上記の例では，「コーヒーか紅茶が好きな人」になります。



和集合

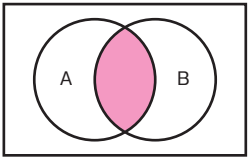
用語

与えられた数をすべて加えることを総和といい，Σ（シグマ）という記号で表されます。
この総和を行う足し算の数が有限ではなく，無限個の演算を行うことを無限級数，または単に級数と呼びます。

発展
その他の集合としては，二つの集合演算をかけ合わせた NAND（NOT ＋ AND），NOR（NOT＋OR）などもあります。
さらに，データベースなどで使用する集合演算には，直積，射影，選択，商などもあります。

②積集合（A AND B，A・B，A∩B，A∧B）

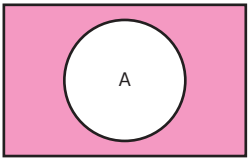
二つの集合の両方に当てはまるものです。前記の例では，「コーヒーも紅茶も好きな人」になります。



積集合

③補集合（NOT A，$\bar{A}$）

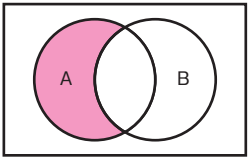
ある集合の否定です。前記の例では，「コーヒーが好きではない人」になります。



補集合

④差集合（A－B）

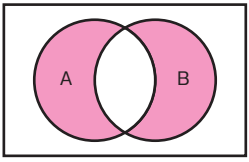
ある集合から，別の集合の条件にあてはまるものを引いたものです。前記の例では，「コーヒーは好きだけれど紅茶は好きではない人」になります。



差集合

⑤対称差集合（A XOR B，A⊕B，A△B）

二つの集合のうち，どちらかの条件にあてはまるものから，両方の条件にあてはまるものを引いたものです。前記の例では，「コーヒーか紅茶が好き，しかし両方は好きではない人」になります。



対称差集合

論理演算

コンピュータの内部表現では，数値のほかに論理（真か偽か）も表現できます。論理演算とは，データを論理で表現し，その組合せを演算することです。

通常，真（TRUE，YES，正しい）を1，偽（FALSE，NO，正しくない）を0で表します。そして，それについてビットごとに

論理和，論理積などを考えることによって，論理演算を行います。

それでは，実際に論理演算を行っていきましょう。

問題

0以上255以下の整数nに対して，

$$\text{next}(n) = \begin{cases} n+1 & (0 \leqq n < 255) \\ 0 & (n = 255) \end{cases}$$

と定義する。next（n）と等しい式はどれか。ここで，x AND y及びx OR yは，それぞれxとyを2進数表現にして，けたごとの論理積及び論理和をとったものとする。

ア （$n+1$）AND 255　　イ （$n+1$）AND 256
ウ （$n+1$）OR 255　　エ （$n+1$）OR 256

（平成22年春 応用情報技術者試験 午前 問1）

解説

next(n) では，演算を行ったときにnの値によって二つの場合に分けて計算するので，それぞれの場合を考えてみましょう。

① $0 \leqq n < 255$の場合

例として，$n = 100$の場合を考えてみましょう。$n = 100$を16ビットの2進数で表すと次のようになります。

00000000 01100100

これに1を加えると$n = 101$となり，最後尾のビットに1を加えるだけなので次のようになります。

00000000 01100101

この$n+1$と同じ演算結果になる選択肢を考えてみましょう。10進数の255は2進数で00000000 11111111，256は2進数で00000001 00000000なので，$n = 100$の場合，次のように

発展

このようなAND演算の使い方を，マスク演算とも呼びます。ビットが1の部分のみを取り出して，ほかの部分をマスク（排除）するときに使用します。
IPアドレスに対するサブネットマスクは，このマスク演算を応用した例です。

なります。

ア　$(n+1)$ AND 255
　　= 00000000 01100101 AND 00000000 11111111
　　= 00000000 01100101（←$n+1$と同じ）

イ　$(n+1)$ AND 256
　　= 00000000 01100101 AND 00000001 00000000
　　= 00000000 00000000（←0になってしまう）

ウ　$(n+1)$ OR 255
　　= 00000000 01100101 OR 00000000 11111111
　　= 00000000 11111111（←255になってしまう）

エ　$(n+1)$ OR 256
　　= 00000000 01100101 OR 00000001 00000000
　　= 00000001 01100101（←357になってしまう）

したがって，アの演算のみ正しい結果となります。

②　$n=255$の場合

選択肢アで，nが255の場合を考えてみます。$n+1=256$，2進数で00000001 00000000をアの計算式で計算します。

$(n+1)$ AND 255
= 00000001 00000000 AND 00000000 11111111
= 00000000 00000000（←0になる）

したがって，$n=255$のときに0になるという条件も満たしています。

≪解答≫ ア

用語

ド・モルガンの法則
$\overline{A \cup B}=\overline{A} \cap \overline{B}$
$\overline{A \cap B}=\overline{A} \cup \overline{B}$

結合の法則
$(A \cup B) \cup C=A \cup (B \cup C)$
$(A \cap B) \cap C=A \cap (B \cap C)$

分配の法則
$A \cap (B \cup C)$
$=(A \cap B) \cup (A \cap C)$
$A \cup (B \cap C)$
$=(A \cup B) \cap (A \cup C)$

論理式の変形・簡略化

論理式では，論理演算の法則・定理を使って変形や簡略化を行うことができます。代表的なものには，ド・モルガンの法則や結合の法則，分配の法則などがあります。

すべての法則を覚えなくても，ベン図などを用いたり，実際の値を入れたりすることで，等価な式を見つけることはできます。それでは，実際の問題を解いてみましょう。

問題

全体集合S内に部分集合AとBがあるとき，$\overline{A} \cap \overline{B}$に等しいものはどれか。ここで，$A \cup B$は$A$と$B$の和集合，$A \cap B$は$A$と$B$の積集合，$\overline{A}$は$S$における$A$の補集合，$A - B$は$A$から$B$を除いた差集合を表す。

ア　$(\overline{A} \cup \overline{B}) - (A \cap B)$　　イ　$(S - A) \cup (S - B)$
ウ　$\overline{A} - B$　　エ　$S - (A \cap B)$

（平成19年秋 ソフトウェア開発技術者試験 午前 問5）

解説

論理演算を行う方法

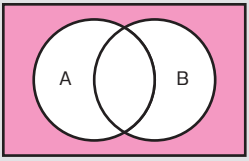
それぞれの選択肢で，すべてのパターンを考えてみます。

A	B	$\overline{A} \cap \overline{B}$	ア $(\overline{A} \cup \overline{B}) - (A \cap B)$	イ $(S - A) \cup (S - B)$	ウ $\overline{A} - B$	エ $S - (A \cap B)$
0	0	1	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	1	0	1
1	1	0	0	0	0	0

$\overline{A} \cap \overline{B}$と同じものは，選択肢ウのみです。

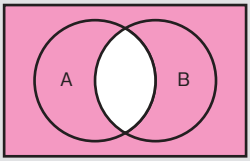
ベン図を書いて解く方法

$\overline{A} \cap \overline{B}$をベン図で書いてみると，以下のようになります。



このベン図は，ウの$\overline{A} - B$を書いたものと同じです。

その他の選択肢は，次のように同じベン図になります。



ア　$(\overline{A} \cup \overline{B}) - (A \cap B)$
イ　$(S - A) \cup (S - B)$
エ　$S - (A \cap B)$

≪解答≫ ウ

覚えよう！

□　けた落ちは有効けた数が減ること，情報落ちは小さい数値の方の情報が落ちること

勉強のコツ

確率・統計のほかに微分積分や指数・対数など，高校レベルの数学も試験範囲です。
たまに出題されますし，基礎としてとても役に立つので，勉強をしたことがない方，苦手な方は勉強しておいた方がいいでしょう。
ただ，労力をかけても出題は1問程度なので，時間がない場合は飛ばしてもOKです。

1-1-2 応用数学

コンピュータでいろいろな処理を行う際に基礎として必要になってくる応用的な数学です。高校から大学の教養過程で学ぶ内容なので完璧に習得しようとすると大変ですが，基本的なことだけは押さえておきましょう。

確率

ある事象（出来事）が起こる確率を求めるときにはまず，**場合の数**を求めます。それぞれの場合が起こる確率が等しいときには，全体の場合の数のうち，ある特定の事象に対しての場合の数の割合が，その事象が起こる確率になります。

$$\text{ある事象が起こる確率} = \frac{\text{ある事象の場合の数}}{\text{全体の場合の数}}$$

それでは，確率に関する問題を解いてみましょう。

問題

Random(n)は，0以上n未満の整数を一様な確率で返す関数である。整数型の変数A，B及びCに対して次の一連の手続を実行したとき，Cの値が0になる確率はどれか。

A＝Random(10)
B＝Random(10)
C＝A－B

ア $\frac{1}{100}$　イ $\frac{1}{20}$　ウ $\frac{1}{10}$　エ $\frac{1}{5}$

（平成20年秋 ソフトウェア開発技術者試験 午前 問4）

解説

A，Bはそれぞれ，Random(10)という関数を使って，0から9までの整数が一様な確率で求められます。つまり，それぞれ10通り｛0，1，2……9｝があるので，その組合せの数は全部で次のようになります。

Aの場合の数(10)×Bの場合の数(10)＝100通り

このうち，C＝A－Bが0になるときは，AとBの値が等しいとき，（A＝0とB＝0，A＝1とB＝1，……A＝9とB＝9）の10通りです。すべての数が出る確率は等しいので，Cの値が0になる確率は次のように求められます。

$$C\text{が}0\text{になる確率} = \frac{C\text{が}0\text{の場合の数}}{\text{全体の場合の数}} = \frac{10\text{通り}}{100\text{通り}} = \frac{1}{10}$$

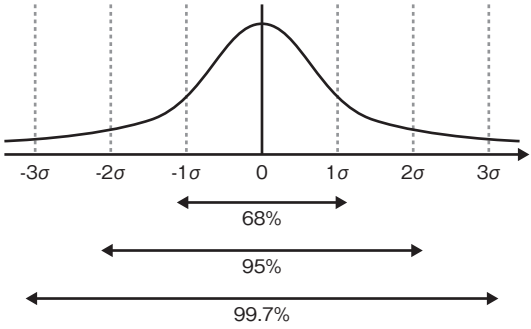
≪解答≫ ウ

統計

統計で一番使われる考え方に，**正規分布**があります。繰り返し実行した場合，それが独立した事象であれば，その分布は正規分布に従うことが知られています。

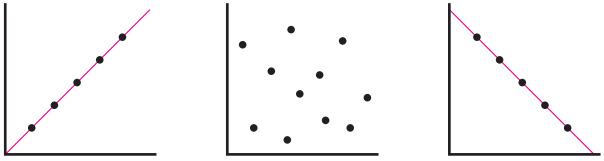
正規分布の場合，その確率の散らばり具合によって，**標準偏差（σ）**が求められます。その分布が正規分布に従っていた場合に，±1σ（－1σから1σ）の間に約68%のデータが含まれます。

±2σの間には約**95%**，±3σの間には約99.7%のデータが含まれます。



正規分布

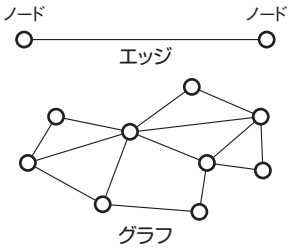
また，データの分布がどれだけ直線に近いかを示す係数に，**相関係数**があります。完全に右上がりの直線上にデータが分布している場合には相関係数が**1**，まったく相関のない無相関な場合には相関係数が**0**になります。さらに，右下がりの直線上にデータが分布している場合には相関係数が**−1**になります。



相関係数=1　相関係数=0　相関係数=−1

グラフ理論

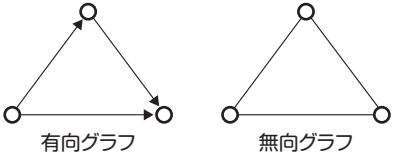
グラフ理論とは数学の1分野で，ノード（node：節点）とエッジ（edge：枝，辺）から構成されるグラフについての理論です。



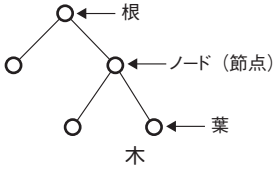
グラフ

関連
グラフ理論は，Facebookでのソーシャルグラフ（友達関係を表すグラフ）や，電車の乗り換え案内などでも利用されています。

グラフには，方向性のある有向グラフと，方向性のない無向グラフの2種類があります。A→Bには行けるがB→Aには行けないことがあるという場合は有向グラフ，どちらからも行けるという場合には無向グラフを用います。



そして，グラフ理論の重要な考え方に**木**という概念があります。木とは，閉路（ループ）を持たないグラフの構造で，根（root）と節点（ノード：node），葉（leaf）を持ちます。



待ち行列理論

待ち行列理論とは，**ある列に並ぶときに，平均でどれだけ待たされるか**を統計学的な計算で求めるための理論です。

列に並んでいるとき，待ち行列のモデルでは次の三つの要素が待ち時間に影響を与えると考えられています。

到着率…… どれくらいの頻度でやって来るか
サービス時間…… 実作業にどれくらい時間がかかるか
窓口の数…… 一つの列に対して窓口がいくつあるか

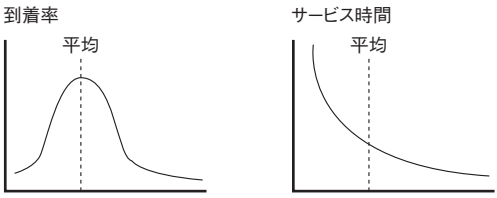
これら三つの要素について，次のようなかたちで待ち行列のモデルを表します。

過去問題をチェック

待ち行列の問題は，応用情報技術者試験 平成22年春 午後 問3，平成21年春 午前 問1，平成24年春 午前 問2などで出題されています。およそ2回に1回は出題される頻出ポイントです。

待ち行列のモデル

M/M/1とは，到着率が一定(D)ではなくランダム(M)(**ポアソン分布**)で，サービス時間が一定(D)ではなくランダム(M)(**指数分布**)な場合の待ち行列のモデルです。



待ち時間の計算

待ち時間は，待ち行列理論を使うと計算できます。待ち行列モデルがM/M/1以外の場合は計算方法が少し複雑なので，あらかじめ計算された表などを利用しますが，M/M/1モデルの場合には次の式で計算できます。

$$\text{利用率}\ \rho = \frac{\text{仕事をしている時間}}{\text{全体の時間}} = \frac{\text{平均サービス時間}}{\text{平均到着間隔}}$$

$$\text{平均待ち時間} = \frac{\rho}{1-\rho} \times \text{平均サービス時間}$$

$$\text{平均応答時間} = \text{平均待ち時間} + \text{平均サービス時間}$$
$$= \frac{1}{1-\rho} \times \text{平均サービス時間}$$

それでは，実際の午後問題を例に，待ち行列の計算を行っていきましょう。

発展

待ち行列モデルの「D」は，確定分布（Deterministic distribution）の意味です。これは，到着率やサービス時間が一定であることを指します。この場合には，ゆらぎがないので待ちが発生しにくくなります。

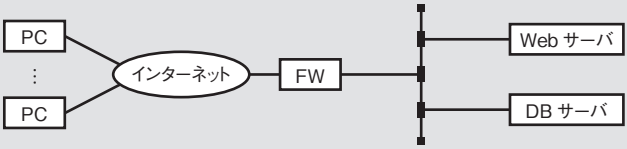
「M」は，到着やサービスがランダムに行われる場合のモデルで，ポアソン分布または指数分布のことを指します。この二つの分布は，着目の仕方の違いで，実は同じ現象を別の視点で見ているのです。例えば，到着の場合，来る確率である到着率はポアソン分布に従いますが，間隔に着目すると到着間隔は指数分布に従います。

試験対策としては，「ポアソン到着率，指数サービス時間」ぐらいで覚えていれば十分ですが，詳しく知りたい方は，確率論を勉強されるのも楽しいと思います。

問題

インターネットを介した情報提供システムに関する次の記述を読んで，設問1，2に答えよ。

Z社は，利用者が希望する映画のタイトル，あらすじ，上映館，上映期間などの映画情報を表示する情報提供サービスを行っており，平均待ち時間の目標値を40ミリ秒以下としている。このサービスに使用する情報提供システムの現在のシステム構成を図1に示す。



FW：ファイアウォール

図1　現在のシステム構成

Webサーバとデータベースサーバ(以下，DBサーバという)を一体のシステムとして，現在のシステムの状況を調査したところ，1分当たりのアクセス数は平均600件，1アクセス当たりの平均処理時間Tpは40ミリ秒であった。また，アクセス頻度はおおむね［ a ］分布に，処理時間はおおむね［ b ］分布に従っていたので，M/M/1の待ち行列モデルによって評価することにした。

設問1　現在のZ社の情報提供システムについて，本文中の［ a ］，［ b ］に入れる適切な字句を答えよ。

設問2　現在のZ社の情報提供システムについて，(1)～(4)に答えよ。ただし，(1)は，整数で答えよ。(2)～(4)は，小数第2位を四捨五入して小数第1位まで求めよ。

(1)　平均到着時間間隔Tr（ミリ秒）を求めよ。

(2)　利用率ρを求めよ。

(3)　平均待ち時間Tw（ミリ秒）を求めよ。

過去問題をチェック

表を用いてM/M/4モデルでの待ち時間を計算する問題は，応用情報技術者試験 平成22年秋 午後 問4で出題されています。
待ち行列の問題は，午後での出題頻度がかなり高いです。

(4) 平均応答時間Ts（ミリ秒）を求めよ。

（平成22年春 応用情報技術者試験 午後 問4抜粋）

解説

設問1

アクセス頻度は，M/M/1モデルでの最初のM，つまり到着率に該当するので，ポアソン分布に従います。処理時間は，2番目のM，つまりサービス時間に該当するので，指数分布に従います。

設問2

(1)

平均到着時間間隔Trを求めます。1分当たりのアクセス数は平均600件なので，何ミリ秒に1回アクセスがあるかという平均到着時間間隔Trは次のようになります。

$$Tr = \frac{\text{時間}}{\text{件数}} = \frac{1[\text{分}] \times 60[\text{秒}] \times 1000[\text{ミリ秒}]}{600[\text{件}]} = 100[\text{ミリ秒}]$$

(2)

利用率ρを求めます。利用率とは，サーバがどれくらいの割合で利用されているかという確率です。平均到着時間間隔Tr＝100［ミリ秒］で，平均サービス時間Tp＝40［ミリ秒］なので，利用率ρは次のようになります。

$$\rho = \frac{\text{仕事をしている時間}}{\text{全体の時間}} = \frac{Tp}{Tr} = \frac{40}{100} = 0.4$$

(3)

平均待ち時間Tw（ミリ秒）を求めます。M/M/1モデルの場合の平均待ち時間の待ち行列の式を適用します。

$$\text{平均待ち時間 Tw} = \frac{\rho}{1-\rho} \times \text{平均サービス時間 Tp}$$

$$= \frac{0.4}{1-0.4} \times 40 = 26.66\cdots\cdots \fallingdotseq 26.7［\text{ミリ秒}］$$

小数第2位を四捨五入して小数第1位まで求めます。

(4)

平均応答時間を求めます。M/M/1モデルの平均応答時間の待ち行列の式を適用します。

$$\text{平均応答時間 Ts} = \frac{1}{1-\rho} \times \text{平均サービス時間 Tp}$$

$$= \frac{1}{1-0.4} \times 40 = 66.66\cdots\cdots \fallingdotseq 66.7\ [\text{ミリ秒}]$$

小数第2位を四捨五入して小数第1位まで求めます。

≪解答≫

設問1　a：ポアソン，b：指数

設問2　(1)100，(2)0.4，(3)26.7，(4)66.7

覚えよう！

- □ **正規分布の場合，±1σの中に入っているデータは全体の約68%，±2σでは約95%**
- □ **平均待ち時間＝ $\frac{\rho}{(1-\rho)}$ ×平均サービス時間**

1-1-3 情報に関する理論

情報技術に応用されている理論です。情報量やオートマトンなど，コンピュータでの処理を効率良く実現するにあたって必要な理論について学びます。

情報量

情報量とは，ある事象が起きたとき，それが**どれくらい起こりにくいか**を表す尺度です。例えば，6月に雨が降るのと雪が降るのとでは雪が降る方が起こりにくいので，情報量としては多くなります。

ある事象が起こるときの情報量を，選択情報量（自己エントロピー）といいます。選択情報量は，次の式で表されます。

$$\text{選択情報量} = -\log_2 P$$

Pは，その事象が起こる確率です。例えば，明日雨が降る確率が50%なら，雨という事象の選択情報量は，

$$雨の選択情報量 = -\log_2 0.5 = -\log_2 \frac{1}{2}$$

$$= -\log_2 2^{-1} = 1$$

となり，選択情報量は1［ビット］となります。

また，系全体，つまり，すべての場合を考えて，その全体の平均をとったときの情報量を**平均情報量**（**エントロピー**）といいます。平均情報量は，次の式で表されます。

$$平均情報量 = \sum_{i=1}^{n} (選択情報量 \times P_i)$$

$$= \sum_{i=1}^{n} \{(-\log_2 P_i) \times P_i\}$$

例えば，明日の天気が，晴れが25%，曇りが25%，雨が50%の可能性だったと考えてみます。晴れと曇りの選択情報量は，$-\log_2 0.25 = -\log_2 2^{-2} = 2$なので，次のようになります。

$$平均情報量 = 2 \times 0.25 + 2 \times 0.25 + 1 \times 0.5 = 1.5$$

したがって，平均情報量は1.5ビットです。

それでは，次の例題を解いてみて，その後で情報量の使い方について考えていきましょう。

問 題

a，b，c，dの4文字からなるメッセージを符号化してビット列にする方法として表のア～エの4通りを考えた。この表はa，b，c，dの各1文字を符号化するときのビット列を表している。メッセージ中でのa，b，c，dの出現頻度は，それぞれ50%，30%，10%，10%であることが分かっている。符号化されたビット列から元のメッセージが一意に復号可能であって，ビット列の長さが最も短くなるものはどれか。

	a	b	c	d
ア	0	1	00	11
イ	0	01	10	11
ウ	0	10	110	111
エ	00	01	10	11

（平成22年秋 応用情報技術者試験 午前 問2）

解説

まず，「符号化されたビット列から元のメッセージが一意に復元可能か」を考える必要があります。

アの場合には，例えばビット列が000のときに，元のメッセージがaaaなのかacなのかcaなのか区別できません。

イも，ビット列が010のとき，baなのかacなのか区別できません。これらの場合には，いくらビット列の長さが短くても採用できません。

エの場合は，すべてが2ビットなので，2ビットごとに区切れば，0110はbcというように一意に復号可能です。しかし，ビット列の長さは2ビット必要になります。

ウの場合，元のメッセージを一意に識別することが可能です。さらに，出現頻度がa，b，c，dそれぞれ50%，30%，10%，10%なので，平均のビット数は，

$$1 \times 0.5 + 2 \times 0.3 + 3 \times 0.1 + 3 \times 0.1 = 1.7 \text{［ビット］}$$

となり，エの場合（2ビット）よりも短くなります。

≪解答≫ ウ

この問題を使って，情報量について計算してみましょう。

a，b，c，dの出現頻度はそれぞれ50%，30%，10%，10%なので，選択情報量を求めると次のようになります。

$$\text{aの選択情報量} = -\log_2 0.5 = -\log_2 2^{-1} = 1$$

$$\text{bの選択情報量} = -\log_2 0.3 = \frac{-\log_{10} 0.3}{\log_{10} 2}$$

$$= \frac{-(\log_{10} 3 - 1)}{\log_{10} 2} \fallingdotseq 1.74$$

$$\text{c，dの選択情報量} = -\log_2 0.1 = \frac{-\log_{10} 0.1}{\log_{10} 2}$$

$$= \frac{1}{\log_{10} 2} \fallingdotseq 3.32$$

ここから，平均情報量（エントロピー）を求めます。

すべての選択情報量を合計して平均情報量を求めると，次のようになります。

平均情報量 = 1 × 0.5 + 1.74 × 0.3 + 3.32 × 0.1+3.32 × 0.1
= 1.686［ビット］

この平均情報量1.686［ビット］というのは系全体が持っている情報量で，この値が実は**圧縮の限界値**になります。前掲の問題で，選択肢ウのようにaを0（1ビット），bを10（2ビット），c，dをそれぞれ110，111（3ビット）割り当てると，平均情報量は1.7と，圧縮の限界値に近い値になります。

オートマトン

オートマトンとは，次のような三つの特徴を持ったシステムのモデルです。

1. 外から，情報が連続して入力される
2. 内部に，**状態**を保持する
3. 外へ，情報を出力する

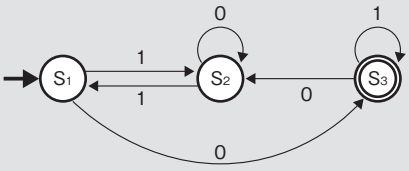
特定の条件が起こったときに，ある状態から別の状態に移ることを遷移といいます。例えばデートのとき，相手が来るのを待っている状態が待ち状態で，相手が来るという遷移条件が起こると，デートをするデート状態に移ります。

オートマトンのうち，状態や遷移の数を有限個で表すことができるものを有限オートマトンといいます。

実際の問題をもとに，オートマトンを体験してみましょう。

問題

次に示す有限オートマトンが受理する入力列はどれか。ここで，S_1は初期状態を，S_3は受理状態を表している。



ア 1011　イ 1100　ウ 1101　エ 1110

（平成21年春 応用情報技術者試験 午前 問3）

解説

初期状態がS_1なので，S_1を起点としてそれぞれの文字列を入力して状態遷移を見ていきます。

ア　1011　$S_1 \xrightarrow{1} S_2 \xrightarrow{0} S_2 \xrightarrow{1} S_1 \xrightarrow{1} S_2$

イ　1100　$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{0} S_3 \xrightarrow{0} S_2$

ウ　1101　$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{0} S_3 \xrightarrow{1} S_3$　受理状態

エ　1110　$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{1} S_2 \xrightarrow{0} S_2$

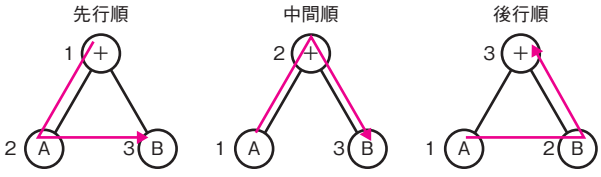
ア，イ，エはS_2，ウのみS_3になります。受理状態とは，その状態で終了すると受理するという状態で，この問題の場合，受理状態はS_3です。

≪解答≫ ウ

このように，初期状態からスタートして順番に状態遷移を行い，最後に受理状態で終われば正常終了というかたちで評価を行うことがオートマトンの基本です。

木の走査順と逆ポーランド記法

グラフ理論における木で，木のそれぞれのノード（節点）を順番に読んでいくことを走査といいます。走査を行う際の順番には，先行順，中間順，後行順の3種類があります。



走査順

先行順，中間順，後行順でノードの内容を表記する方法をそれぞれ，前置表記法（ポーランド記法），中置表記法，後置表記法（逆ポーランド記法）といいます。

人間の頭脳が理解しやすいのは中置表記法ですが，コンピュータは後置表記法（逆ポーランド記法）の方が処理しやすいという違いがあります。そのため，演算などを行うときには，コンピュータ内で逆ポーランド記法に置き換えていきます。

以下の問題で，実際に表記法を置き換えてみましょう。

問題

後置表記法（逆ポーランド表記法）では，例えば，式 $Y=(A-B)\times C$ を $YAB-C\times=$ と表現する。

次の式を後置表記法で表現したものはどれか。

$$Y=(A+B)\times(C-(D\div E))$$

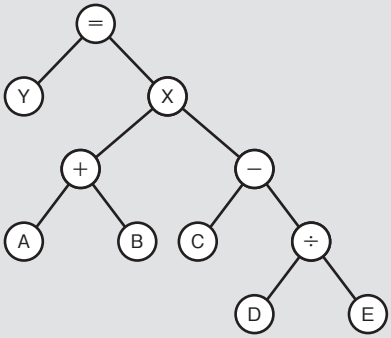
ア　$YAB+C-DE\div\times=$　　イ　$YAB+CDE\div-\times=$

ウ　$YAB+EDC\div-\times=$　　エ　$YBA+CD-E\div\times=$

（平成22年秋 応用情報技術者試験 午前 問1）

解説

問題文の式は中置表記法なので，まず次のように木構造に置き換えます。



この木を逆ポーランド記法にするには後行順で順に読んでいけばいいので，次のようになります。

$YAB + CDE \div - \times =$

≪解答≫ イ

BNF（Backus-Naur Form）記法

BNFは，文法などの形式を定義するために用いられる言語で，プログラミング言語などの定義に利用されています。繰返しの表現に再帰を使うのが特徴です。

実際の問題を例に，BNFについて学習していきましょう。

過去問題をチェック

BNFのアルゴリズムに関する問題は，応用情報技術者試験 平成22年秋 午後 問2で出題されています。

問題

あるプログラム言語において，識別子（identifier）は，先頭が英字で始まり，それ以降に任意個の英数字が続く文字列である。これをBNFで定義したとき，aに入るものはどれか。

<digit> ::＝0｜1｜2｜3｜4｜5｜6｜7｜8｜9
<letter> ::＝A｜B｜C｜…｜X｜Y｜Z｜a｜b｜c｜…
｜x｜y｜z
<identifier> ::＝ [a]

ア <letter>｜<digit>｜<identifier><letter>｜<identifier><digit>
イ <letter>｜<digit>｜<letter><identifier>｜<identifier><digit>
ウ <letter>｜<identifier><digit>
エ <letter>｜<identifier><digit>｜<identifier><letter>

（平成23年特別 応用情報技術者試験 午前 問4）

解説

記号の「::=」は「定義する」，「|」は「いずれか」を意味します。したがって，

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

は，<digit>が0から9までの数字であることを定義し，

<letter> ::= A | B | C | … | X | Y | Z | a | b | c | … | x | y | z

は，<letter>が大文字か小文字の英字であることを定義しています。

識別子<identifier>については，問題文に「先頭が英字で始まり，それ以降に任意個の英数字が続く文字列」と記載されています。つまり，識別子<identifier>は，

<identifier> ::= <letter> …最初の1文字（英字）のみか，そこに再帰を用いて，

<identifier> ::= <identifier><digit>

または，

<identifier> ::= <identifier><letter>

というかたちで，2番目以降には英数字（英字か数字のどちらか）が続くという状態を表現できます。したがって，次のようになります。

<identifier> ::= <letter> | <identifier><digit> | <identifier><letter>

≪解答≫ エ

計算量（オーダー）

計算量とは，あるプログラム（アルゴリズム）を実行するのにどれくらいの時間がかかるかを，入力データに対する増加量で表したものです。計算量を表すときには，***O*-記法**という表記法で，*O*（オーダー）という考え方が用いられます。

例えば，入力データの数（n）が増加したとき，その計算量もnに比例して増加していくときのことを，*O*（n）と表します。n^2に比例して増加する場合には，*O*（n^2）です。オーダーは，nが非常に大きいときの計算量を考えるので，数値が小さい場合には無視し，定数も無視します。例えば，入力データの数（n）が増加し

たとき，$3n^2 + n + 2$に比例して計算量が大きくなるときには，nが大きくなってもあまり増加しない$n + 2$の部分や，比例定数3の部分は無視されて，$O(n^2)$となります。

以下に，試験によく出てくる代表的なO（オーダー）とその例を示します。

O（オーダー）	例（アルゴリズム）
$O(1)$	ハッシュ
$O(\log n)$	二分探索
$O(n)$	線形探索
$O(n\log n)$	クイックソート，シェルソート
$O(n^2)$	バブルソート，挿入ソート

覚えよう！

- □ **木の走査順は，先行順，中間順，後行順の３種類。後行順で読むと，逆ポーランド記法が表記できる**
- □ **二分探索の計算量は$O(\log n)$，クイックソートの計算量は$O(n\log n)$**

1-1-4 通信に関する理論

情報理論の中でも，特に通信に関する理論です。データがネットワークを通じて伝送されるときには途中で誤りが発生することが多く，その誤りの検出・訂正がこの分野のカギになります。

誤り検出・訂正

二つの装置間で通信を行うとき，通信時の回線状況や機器同士のやり取りなど，様々な原因で通信データの送信誤りが起こります。誤り検出・訂正の種類とその代表的な手法を以下の表にまとめます。

誤り検出・訂正の種類	代表的な手法
1ビット誤り検出	パリティ（奇数パリティ，偶数パリティ）
1ビット誤り訂正＋1ビット誤り検出	垂直パリティ＋水平パリティ
1ビット誤り訂正＋2ビット誤り検出	ハミング符号
nビット誤り検出	CRC

発展

メモリのエラーなど，めったに誤りが起こらず，起こっても1ビットのみのことが多い場合には，誤り訂正も可能なハミング符号が用いられます。ハミング符号による誤り訂正ができるメモリのことを，ECCメモリ（Error Check and Correct Memory）といい，サーバなど，信頼性が求められる場面で使用されます。

逆に，通信データのように，頻繁に誤りが起こり，また一度にまとめて起こること（バースト誤り：P.50「発展」参照）が多い場合には，複数ビットの誤りを検出できるCRCを用います。

それでは，それぞれの代表的な手法について詳しく見ていきましょう。

パリティ

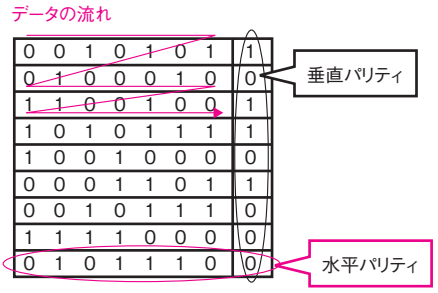
パリティとは，ある数字の並びの合計が偶数か奇数かによって通信の誤りを検出する技術です。データの最後にパリティビットを付加し，そのビットを基に誤りを検出します。

そのとき，**1の数が偶数になるように**パリティビットを付加するのが**偶数パリティ**，**奇数になるように**パリティビットを付加するのが**奇数パリティ**です。

例えば，7ビットのデータ1100010を考えます。

偶数パリティの場合，現在のデータは1の数が奇数なので最後に1を付加して11000101とします。奇数パリティの場合は，すでに1の数が奇数なので最後に0を付加し，11000100とします。

また，パリティは，使い方によっては，誤り訂正を行うことができます。データが送られるごとに，最後に1ビットのパリティビットを付けるやり方を**垂直パリティ**といいます。7ビットデータを送り，それに1ビット垂直パリティを加える，ということを何度か繰り返し，データを送信します。そして最後に，それぞれのビットのデータを横断的に見て，全体で誤りがないかどうかをチェックするのに**水平パリティ**を用います。図に表すと以下のようになります。



垂直パリティと水平パリティ

垂直パリティと水平パリティを組み合わせることにより，どこかに1ビットの誤りが発生した場合にその場所を特定でき，ビットを反転させることでエラーを訂正できます。

ハミング符号

ハミング符号とは，データにいくつかの冗長ビットを付加することによって，1ビットの誤りを検出し，それを訂正できる仕組みです。実際の問題を例に誤り箇所を検出し，それの訂正を行ってみましょう。

問題

符号長7ビット，情報ビット数4ビットのハミング符号による誤り訂正の方法を，次のとおりとする。

受信した7ビットの符号語 $x_1x_2x_3x_4x_5x_6x_7$（x_k ＝0又は1）に対して

$$c_0 = x_1 + x_3 + x_5 + x_7$$
$$c_1 = x_2 + x_3 + x_6 + x_7$$
$$c_2 = x_4 + x_5 + x_6 + x_7$$

（いずれもmod2での計算）

を計算し，c_0，c_1，c_2 の中に少なくとも一つは0でないものがある場合には，

$i = c_0 + c_1 \times 2 + c_2 \times 4$

を求めて，左からiビット目を反転することによって誤りを訂正する。

受信した符号語が1000101であった場合，誤り訂正後の符号語はどれか。

ア　1000001　　イ　1000101
ウ　1001101　　エ　1010101

（平成23年秋 応用情報技術者試験 午前 問3）

解説

この問題のハミング符号は，4ビットから成るデータに3ビットの冗長ビットを加えて7ビットにしたものです。このハミング符号では，正しくデータが送られた場合には，c_0，c_1，c_2 の演算結果はすべて0になるはずです。

ここで，受信した符号語1000101について，c_0，c_1，c_2の演算を行ってみると，

$c_0 = x_1 + x_3 + x_5 + x_7 = 1 + 0 + 1 + 1 = 3 \rightarrow 3 \bmod 2 = 1$

$c_1 = x_2 + x_3 + x_6 + x_7 = 0 + 0 + 0 + 1 = 1 \rightarrow 1 \bmod 2 = 1$

$c_2 = x_4 + x_5 + x_6 + x_7 = 0 + 1 + 0 + 1 = 2 \rightarrow 2 \bmod 2 = 0$

となり，誤りがあることが分かります。どこのビットが誤りかは，iを求める式より次のように導き出すことができます。

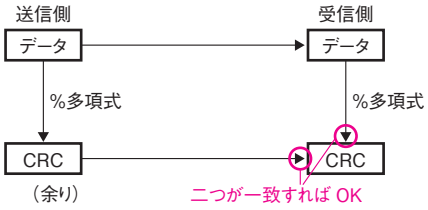
$i = c_0 + c_1 \times 2 + c_2 \times 4 = 1 + 1 \times 2 + 0 \times 4 = 3$

3ビット目が誤っていることが分かるので，これを修正します。したがって，訂正後のハミング符号は，1010101となります。

《解答》エ

CRC（Cyclic Redundancy Check）

CRCは，連続する誤りを検出するための誤り制御の仕組みです。送信する元となるデータを，あらかじめ決められた多項式で除算し，その余りをCRCとします。CRCのエラーチェックは，以下の図のように示すことができます。



CRCのエラーチェック

なお，CRCのエラーチェックは，改ざん防止などのセキュリティチェックには使えません。これは，データは暗号化されていないテキストであり，多項式は公開されているため，改ざんして再計算することが可能だからです。

発展

通信データの場合，ケーブルの不良や混線などによって誤りが一度にまとめて起こることがあります。こういった誤りをバースト誤りと呼びます。
バースト誤りに対しては，複数ビットの誤りを検出できるCRCを用います。
家庭や社内のLANで一般的に使われているイーサネットのパケットでは，誤り検出にCRCを用いています。

過去問題をチェック

CRCについては，応用情報技術者試験 平成21年秋 午前 問4で出題されています。

覚えよう！

- □ 1の数を足すと奇数になるのが奇数パリティ，偶数になるのが偶数パリティ
- □ ハミング符号では，1ビットの誤り訂正ができる
- □ CRCでは，バースト誤りが検出できる

1-1-5 計測・制御に関する理論

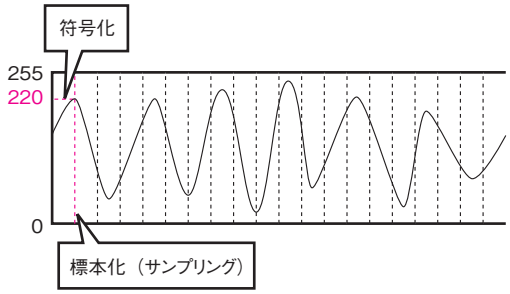
計測・制御に関する理論は，主に組込系のシステムにおいて使われます。

A/D変換，D/A変換

人間世界にあるアナログの情報（音，画像など）をコンピュータで扱うためには，ディジタルのデータに変換する必要があります。この変換を**A/D変換**（Analog/Digital変換）といいます。また，そのディジタルのデータを実際に用いる場合（録音したディジタル音声を聞く場合など）は，逆にアナログの情報に変換する必要があります。この逆の変換を**D/A変換**（Digital/Analog変換）といいます。

A/D変換を行うためには，**標本化**と**符号化**という二つの作業が必要です。標本化とは，連続のデータを一定の間隔をおいてサンプリングすることで，符号化は，そのサンプリングしたデータをディジタルに置き換えることです。

例えば，下図のような音の波があったときに，一定間隔でデータを取得することを標本化，それをディジタルのデータに置き換えることを符号化といいます。



標本化と符号化

過去問題をチェック

A/D変換やD/A変換は，応用情報技術者試験 平成22年春 午前 問23，平成22年秋 午前 問13，平成23年秋 午前 問4で出題されています。また，PCMやサンプリングについても，応用情報技術者試験 平成22年春 午前 問27，平成22年秋 午前 問3に出題されています。
アナログ／ディジタルの相互変換は，隠れた出題ポイントです。

発展

音を標本化し，符号化したデータを格納する方式で代表的なものにPCM（Pulse Code Modulation）があります。音楽CDやISDNなどでは，PCMが用いられています。

それでは，次の問題を解いてみましょう。

問 題

サンプリング周波数40kHz，量子化ビット数16ビットでA/D変換したモノラル音声の1秒間のデータ量は，何kバイトとなるか。ここで，1kバイトは1,000バイトとする。

ア　20　　イ　40　　ウ　80　　エ　640

（平成23年秋 応用情報技術者試験 午前 問4）

解 説

サンプリング周波数40kHzということは，1秒間に40×10^3回サンプリングするということです。そのため，1サンプリング当たりの量子化ビット数を16ビットでA/D変換した場合の1秒間のデータ量は，以下のようになります。

$16［ビット］\times 40 \times 10^3［回／秒］\div 8［ビット／バイト］$
$= 80 \times 10^3［バイト］= 80［kバイト］$

≪解答≫ウ

標本化定理

標本化定理とは，ある周波数のアナログ信号をディジタルデータに変換するとき，それをアナログ信号に復元するためには，その周波数の**2倍のサンプリング周波数**が必要だという定理です。例えば，4kHzまでの音声データを復元させるためには，その倍の8kHzのサンプリング周波数が必要です。

実際の問題で考えてみましょう。

問 題

0～20kHzの帯域幅のオーディオ信号をディジタル信号に変換するのに必要な最大のサンプリング周期を標本化定理

によって求めると，何マイクロ秒か。

ア 2.5　　イ 5　　ウ 25　　エ 50

（平成21年秋 応用情報技術者試験 午前 問3）

解 説

最大で20kHzのオーディオ信号なので，必要なサンプリング周波数を標本化定理によって求めると，次のようになります。

$20\text{kHz} \times 2 = 40\text{kHz}$

ここからサンプリング周期を求めます。

$$\text{サンプリング周期} = \frac{1}{\text{サンプリング周波数}} = \frac{1}{40000}$$

$$= 0.000025\text{［秒］} = 25\text{［マイクロ秒］}$$

《解答》ウ

制御システム

制御システムとは，ロボットや機械など，他の機器を制御するシステムです。制御システムを構成する要素には以下のようなものがあります。

①センサ

動きや温度などを**計測する**ための機構です。センサには，ロボットや機械自身の状態を知るための内界センサと，周りの状況を知るための外界センサがあります。

② アクチュエータ

機械・機構を**物理的に動かす**ための機構です。制御棒などを実際に動かしたり，ロボットの腕を動かしたりします。

過去問題をチェック
アクチュエータについては，応用情報技術者試験 平成22年秋 午前 問4で出題されています。

覚えよう！

- □ **標本化定理では，2倍のサンプリング周波数が必要**
- □ **アクチュエータは，実際の機械を物理的に動かす**

1-2 アルゴリズムとプログラミング

アルゴリズムとプログラミングは，午前でも午後でも出題される，テクノロジ系では最重要分野です。『データ構造＋アルゴリズム＝プログラム』という有名な古典があるほど，データ構造とアルゴリズムはプログラミングのカギになります。データ構造とアルゴリズムの定番を押さえて，様々なプログラミング言語を学習していきましょう。

勉強のコツ

プログラミングは，“慣れ”が一番大切なので，基本を押さえたあとは演習を行いましょう。午後のアルゴリズム問題を数多く演習するか，または実際にプログラミングしてみるなどして，実践力をつけると効果的です。

発展

データ構造は，プログラムの様々な部分で使われるため，いったん決めると変更が大変です。
そして，データ構造の選び方によって，処理速度が変わってきたり，変更のしやすさが変わったりと，影響が大きい部分でもあります。そのため，「データ構造の選び方」が，プログラマの腕の見せ所になります。

発展

C++，Javaなどは基本的には静的配列ですが，ライブラリを使用することで動的配列にすることができます。Perlなどでは，言語に組み込まれており，意識せず可変長の動的配列を使うことが可能です。

1-2-1 データ構造

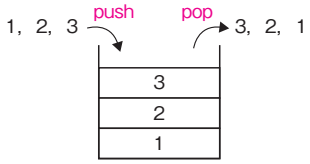
データ構造とは，データの集まりをコンピュータのメモリ上でどのように保持するかを決めた形式です。数値型，文字型，浮動小数点型など，基本的なデータ構造のほかに，配列，スタック，リスト，キュー，木，セマフォなど，様々なデータ構造があります。

配列

データ構造を複数個連続させたものです。複数のデータを一度に管理することができます。例えば，文字型を連続させて文字配列とすることで，文字列を表現することができます。同じデータ型のデータをあらかじめ決めた数しか収納できない**静的配列**が基本ですが，最近のプログラム言語では，異なった型のデータを収納することや，データの数に応じて可変長の配列とすることが可能な**動的配列**を扱えるものもあります。

スタック

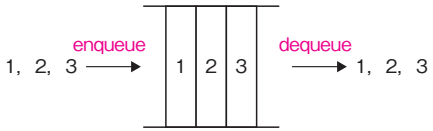
スタックとは，**後入れ先出し**（**LIFO**：Last In First Out）のデータ構造です。データを取り出すときには，最後に入れたデータが取り出されます。スタックにデータを入れる操作を**push**操作，データを取り出す操作を**pop**操作と呼びます。スタックは，CPUのレジスタやプログラムでの関数呼出しなど，現在のコンピュータシステムで非常に広範囲で使われています。



スタック

キュー（待ち行列）

キュー（待ち行列）とは，**先入れ先出し**（**FIFO**：First In First Out）のデータ構造です。データを取り出すときには，最初に入れたデータが取り出されます。キューにデータを入れる操作を**enqueue**操作，データを取り出す操作を**dequeue**操作と呼びます。キューは，プリンタの出力やタスク管理など，順番どおりに処理する必要がある場合に用いられます。データに優先度をつけ，優先度を考慮して順番を決定する**優先度付きキュー**もよく用いられます。



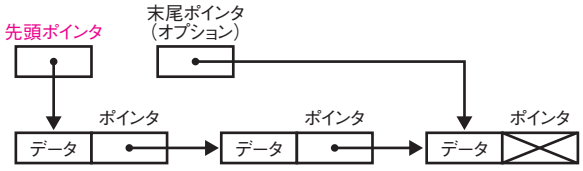
キュー

リスト

リストは，線形リストともいい，順序づけられたデータの並びです。データ構造としては，データそのものを格納するデータ部と，データの並び（次のデータへのポインタ，前のデータへのポインタなど）を格納するポインタ部を合わせて管理します。データの先頭を指し示すために，**先頭ポインタ**を使用し，管理します。また，先頭ポインタだけでなく，**末尾ポインタ**を用いて，最後方のデータに簡単にたどりつけるようにすることもあります。

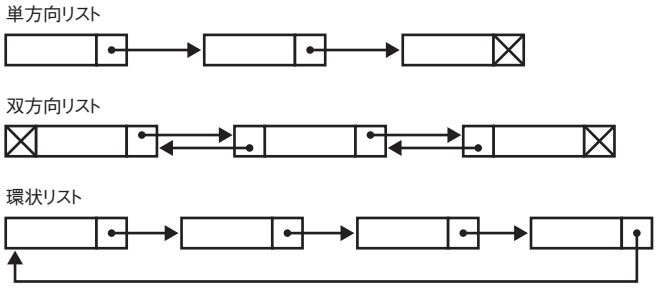
過去問題をチェック

リストについての問題は，応用情報技術者試験 平成21年秋 午前 問5，平成22年秋 午前 問5で問われています。
午後問題でも，平成21年春 午後 問2で単方向リストが，平成22年春 午後 問2で双方向リストが出題されています。
データ構造の中では，リストが応用情報技術者試験の最頻出項目です。



データ部とポインタ部

リストには，次のデータへのポインタのみをもつ**単方向リスト**と，前へのポインタと次へのポインタをもつ**双方向リスト**があります。また，最後尾のデータから先頭に戻って環状につなげる**環状リスト**などもあります。



リストの種類

実際の問題を例に，リストの動きを確認してみましょう。

問 題

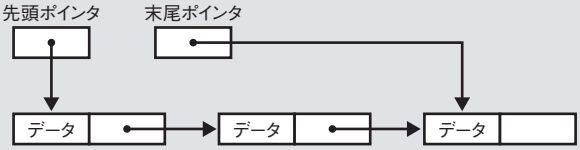
先頭ポインタと末尾ポインタをもち，多くのデータがポインタでつながった単方向の線形リストの処理のうち，先頭ポインタ，末尾ポインタ又は各データのポインタをたどる回数が最も多いものはどれか。ここで，単方向のリストは先頭ポインタからつながっているものとし，追加するデータはポインタをたどらなくても参照できるものとする。

ア　先頭にデータを追加する処理
イ　先頭のデータを削除する処理
ウ　末尾にデータを追加する処理
エ　末尾のデータを削除する処理

（平成22年秋 応用情報技術者試験 午前 問5）

解説

先頭ポインタと末尾ポインタをもち，多くのデータがポインタでつながった単方向の線形リストを図で表すと，以下のようになります。



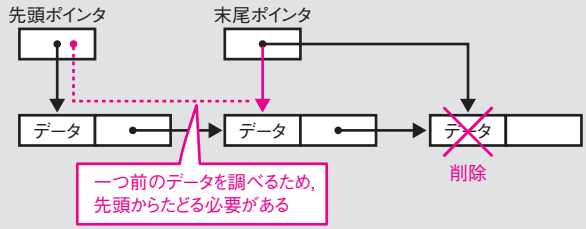
この線形リストに，ア，イ，ウ，エそれぞれの処理を行う場合を考えてみます。

ア　先頭にデータを追加する処理は，現在の先頭ポインタの値を追加するデータの次へのポインタにし，先頭ポインタが追加するデータを指し示すようにすればOKです。

イ　先頭のデータを削除する処理は，先頭ポインタを2番目のデータ（＝1番目のデータの次へのポインタ）を指すようにすればOKです。

ウ　末尾にデータを追加する処理は，末尾ポインタをたどり，末尾のデータの次へのポインタが，追加するデータを指し示すようにすればOKです。

エ　末尾のデータを削除する処理は少し複雑です。末尾のデータを削除すると，末尾ポインタは**末尾の一つ前のデータ**を指し示す必要があるのですが，それを見つけるためには，先頭から末尾の直前までたどっていく必要があります。図にすると，次のようなかたちです。

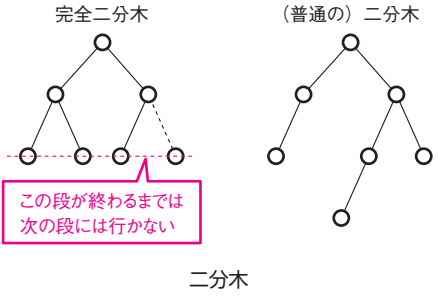


先頭から順に末尾の一つ前までたどるには，(データの個数－1)回，データのポインタをたどる必要があります。したがって，選択肢エが，ポインタをたどる回数が最も多くなります。

≪解答≫エ

木

木(木構造)とは，グラフ理論(「1-1-2　応用数学」P.34参照)の木の構造をしたデータ構造です。ノード間の関連は親子関係で表され，根ノード以外の子ノードでは，親ノードは必ず一つです。親ノードに対する子ノードの数が二つまでに限定されるものを**二分木**，三つ以上もてるものを**多分木**と呼びます。また，二分木のうち形が完全に決まっているもの，つまり，一つの段が完全にいっぱいになるまでは次の段に行かないものを**完全二分木**といいます。

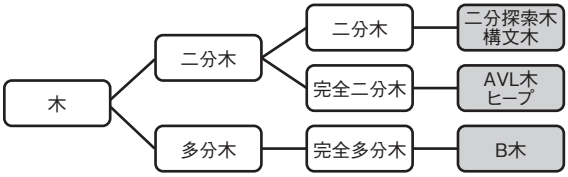


二分木の実用例としては，データの大小関係を，木を使ってたどっていく**二分探索木**や，構文や文法を表現する**構文木**などがあります。完全二分木の実用例としては，二分探索木を完全

過去問題をチェック

木についての問題は，ソフトウェア開発技術者試験(平成20年以前)ではよく出題されていて，平成20年秋 午前 問9(B木)と問10(葉の数)，平成20年春 午前 問9(二分木)など，ほぼ毎回問われていました。ただ，応用情報技術者試験になってから(平成21年以降)は，難易度が高いせいか出題頻度が下がっています。応用情報技術者試験 平成23年特別 午前 問6では出題されていますし，いちおう押さえておいた方がいい分野ですが，難しければ飛ばしてもOKです。

二分木に変換したAVL木や，根から葉に向けてだけデータを整列させたヒープなどがあります。多分木の例としては，完全多分木で二分探索木の多分木バージョンであるB木などがあります。まとめると，以下のようなかたちになります。



木構造の分類

それでは，実際の問題を例に，木の表現方法を確認してみましょう。

問題

節点の集合が{1，2，…，*n*}である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*の親の節点を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。

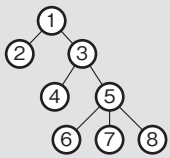
i	1	2	3	4	5	6	7	8
A[*i*]	0	1	1	3	3	5	5	5

ア　1　　イ　3　　ウ　5　　エ　7

（平成20年秋 ソフトウェア開発技術者試験 午前 問10）

解説

配列Aを見ると，節点1が根で，節点2，3の親の節点は1，節点4，5の親の節点は3，節点6，7，8の親の節点が5ということが分かります。図にすると，右図のようなかたちです。



葉とは，子のない節点のことです。図より，2，4，6，7，8の5つの節点は葉になるので，木の葉の数は5つです。

≪解答≫ウ

● ヒープ

ヒープとは，完全二分木の一種で，最大値または最小値を求めるのに適したデータ構造です。「**子要素は親要素より常に大きいか等しい**」か，または逆に「**子要素は親要素より常に小さいか等しい**」のどちらかになります。子要素が親要素より常に大きいか等しい場合，根の値は全体の最小値，逆の場合は最大値になります。

実際の問題を例に，ヒープの性質を確認していきましょう。

問題

配列内に構成されたヒープとして適切なものはどれか。

ア

添字	1	2	3	4	5	6	7	8	9	10	11
要素	1	3	5	12	6	4	9	15	14	8	11

イ

添字	1	2	3	4	5	6	7	8	9	10	11
要素	1	5	3	12	6	4	9	15	14	8	11

ウ

添字	1	2	3	4	5	6	7	8	9	10	11
要素	1	5	3	12	8	4	9	15	14	6	11

エ

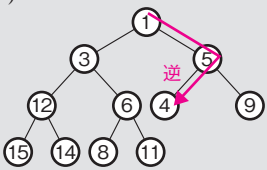
添字	1	2	3	4	5	6	7	8	9	10	11
要素	1	6	3	12	5	4	9	15	14	8	11

（平成18年秋 ソフトウェア開発技術者試験 午前 問9）

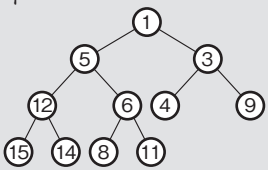
解説

ヒープは完全二分木で，形が完全に決まっています。そのため，単純に配列で表すことが可能です。その場合，1が根，2，3がその子，4，5が2の子，6，7が3の子……という順番にデータを構成していきます。そのルールを使って選択肢のア，イ，ウ，エを木構造で表現すると，以下のかたちになります。

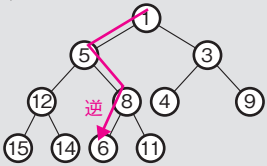
ア



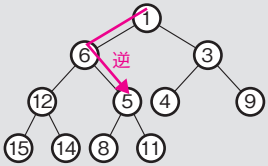
イ



ウ



エ



根から葉までのルートだけをヒープのルール，今回の場合は「子要素は親要素より常に**大きい**か等しい」の条件でチェックします。すると，アの場合は，1→5→4のルートで5と4が逆になっています。イは，すべてのルートで条件を満たしています。ウの場合は，1→5→8→6のルートで8と6が逆になっています。エの場合は，1→6→5のルートで6と5が逆になっています。

≪解答≫イ

覚えよう！

- □ **リストには単方向リストと双方向リストがあり，単方向だと後ろからたどれないので削除時の探索量が多くなる**
- □ **ヒープは完全二分木の一種で，子要素は親要素より「常に大きいか等しい」または，「常に小さいか等しい」**

勉強のコツ

アルゴリズム分野のポイントは，定番アルゴリズムについてしっかり知っておくことと，プログラムを読み解くスキルを身に付けることの二つです。
午前問題では，定番アルゴリズムについての知識問題と，簡単な流れ図を読み解く問題が中心になります。

発展

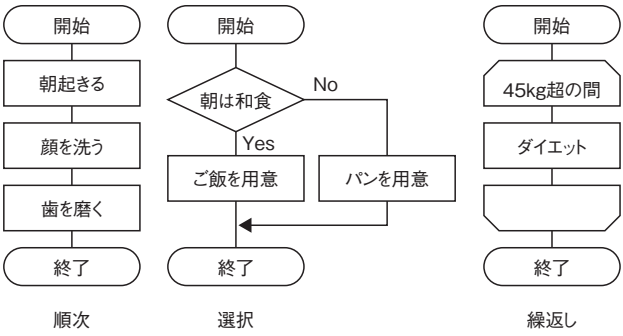
アルゴリズムは，業務でプログラムを作成するときにも使えます。自分のやりたいことを実現してくれるアルゴリズムを「アルゴリズム辞典」などから探し，それを適用することで，自分で一から考えなくても効率的なプログラムを組むことができます。

1-2-2 アルゴリズム

アルゴリズムとは，処理の流れ，手順を記述したものです。プログラムからプログラミング言語に依存する部分を取り除き，処理の流れや変数などの骨組みだけを表したものがアルゴリズムになります。探索や整列などの定番アルゴリズムを知ることで，プログラミングの手法や計算量の考え方など，プログラミングに必要な基礎力を身に付けることができます。

流れ図（フローチャート）

流れ図（フローチャート）では，順次・選択・繰返し（ループ）という基本3構造（ダイクストラが提言したので，**ダイクストラの基本3構造**とも呼ばれます）を用いて，プログラムの骨組みを記述します。



流れ図を基本3構造で記述

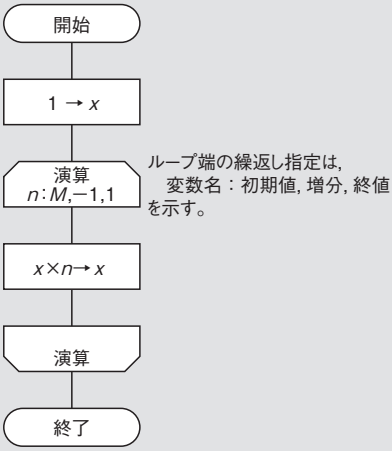
応用情報技術者試験の午後では，疑似言語を用いて基本3構造を表現することが多いのですが，このとき，選択を**if**，繰返しを**while**または**for**としてアルゴリズムを表現します。

また，アルゴリズムでは，プログラミングを1行1行追って確認していくことを**トレース**といいます。変数の値を1行ごとに確認していくことが大切です。

それでは，流れ図の例として，次の問題を解いてみましょう。

問題

正の整数Mに対して，次の二つの流れ図に示すアルゴリズムを実行したとき，結果xの値が等しくなるようにしたい。aに入れる条件として，適切なものはどれか。



ア　$n<M$　　イ　$n>M-1$

ウ　$n>M$　　エ　$n>M+1$

（平成22年秋 応用情報技術者試験 午前 問7）

解説

二つの流れ図では，左が繰返し，右が選択を用いています。このように，選択を用いることで，繰返しと同じような内容を表現することができます。

まず左の流れ図（繰返し）では，ループの繰返し指定で，nの初期値がM，増分が-1，終値が1となっています。このループの間，$x \times n \rightarrow x$という計算を繰り返します。xの初期値は1なので，ここでの計算結果は以下のかたちになります。

$x = 1 \times \boldsymbol{M} \times (M-1) \times (M-2) \times (M-3) \times \cdots\cdots \times 3 \times 2 \times 1$

右の流れ図（選択）で表現されるループは，nの初期値が1でスタートし，$n+1 \rightarrow n$でnを1ずつ増やしながら，$x \times$

過去問題をチェック

流れ図を読み解く問題は，応用情報技術者試験 平成22年春 午前 問5，平成22年秋 午前 問7，平成23年特別 午前 問9，平成23年秋 午前 問8と，ほぼ毎回出題されています。午後問題では，疑似言語になりますが，午後問2で毎回，プログラムの流れを読み解く問題が出されます。

勉強のコツ

流れ図やプログラムを読み解く問題では，面倒がらずに一つ一つをトレースすることが大切です。慣れないうちは大変かもしれませんが，変数の値の変化を1行1行確かめながら追っていくと，確実に答えを導き出せるようになります。

$n \rightarrow x$という計算を繰り返します。xの初期値は1なので，ここでの計算結果は次のようになります。

$x = 1 \times 1 \times 2 \times 3 \times \cdots\cdots$

ここで，左図のループとxの値が等しくなるためには，最後をMで終わらせて以下のようにする必要があります。

$x = 1 \times 1 \times 2 \times 3 \times \cdots\cdots \times (M-3) \times (M-2) \times (M-1) \times \boldsymbol{M}$

つまり，nがMまでの間は計算して，Mを超えたら終了となればOKです。したがって，空欄aに入れる条件は$\boldsymbol{n > M}$としてnがMを超えたら終了というかたちにすれば，左の流れ図と同じ結果になります。

≪解答≫ウ

勉強のコツ

定番のアルゴリズムでは，同じ結果を出すために複数の方法が存在します。それぞれの手法には特徴があり，得意な条件とそうでない条件があります。そのアルゴリズムにかかる計算量などを中心に，やり方だけでなくそれぞれの特徴を押さえておきましょう。

探索アルゴリズム

探索のアルゴリズムは，データの並びの中から目的のものを見つけ出すという最も基本的な定番アルゴリズムです。探索アルゴリズムの代表的なものには，以下の三つがあります。

①線形探索

データを先頭から順番に探索していく単純なアルゴリズムです。データがランダムに並んでいる場合，探索するデータは先頭にあることも最後尾にあることもありますが，平均すると大体真ん中くらいで見つかると考えられます。そのため，n個のデータで探索を行うと，平均探索回数はn／2回，**計算量は$O(n)$**となります（計算量については，P.46を参照）。

②2分探索

データをあらかじめ整列させておき，最初に真ん中のデータと探索するデータを比較します。二つのデータの関係から前後どちらのグループに目的のデータがあるかを予測し，そのグループの真ん中のデータと比較します。例えば，次のようなデータがある場合を考えます。

探索されるデータ

1	2	3	4	5	6	7	8	9

探索するデータ

3

最初に，真ん中のデータ「5」と探索するデータ「3」を比較します。5＞3なので，探索するデータは「5」より前のグループ

1	2	3	4

の中にあることが分かります。したがって，このグループについて再度，真ん中の値を求めます。このとき，「2」と「3」はどちらも真ん中なのでどちらでもいいのですが，通常は前の値「2」をとります。このとき，2＜3なので，探索するデータは「2」より後ろにあることが分かります。その後ろのグループ「3」「4」の真ん中のデータ「3」と比較し，3＝3でデータが見つかります。

このように，半分にデータを絞って探索を行うため，n回の探索で2^nまでのデータ数に対応できます。したがって，**計算量は$O(\log n)$**となります。

③ハッシュ表探索

ハッシュ関数を利用し，データからハッシュ値を求めることによって探索します。例えば，ハッシュ関数として，

h（x）＝ x ％ 5（％は余りを計算する演算子）

というものを設定し，データの格納場所を五つ用意します。

ハッシュ表の例

ハッシュ値	データ
0	25
1	11
2	7
3	13
4	4

ここで，探索するデータが「7」のとき，h（7）＝7 ％ 5＝2となり，ハッシュ値が「2」の場所を見るとデータ「7」が見つかります。この方法は演算ですぐに格納場所が見つかるので，データ量に

発展

O-記法での対数関数では，$\log_2 n$や$\log_{10} n$などといったかたちの底は記入せず，$O(\log n)$と表現します。

O-記法は，「だいたいこれくらいの割合で増加する」ということを示す記法です。対数関数の場合，底がいくつでも増加の割合は同じなので，底を記入する必要がないのです。

発展

シノニムの解決方法としては，そのシノニムデータを線形リストによって管理するチェイン法や，次の空き位置にデータを格納するオープンアドレス法などがあります。チェイン法については，応用情報技術者試験 平成21年春 午後 問2で詳しく出題されています。

過去問題をチェック

ハッシュ表探索に関するプログラミング問題が，応用情報技術者試験 平成21年春 午後 問2，平成23年秋 午後 問2で出題されています。また午前でも，ハッシュ法について，応用情報技術者試験 平成23年特別 午前 問8，平成23年秋 午前 問5でも出題されています。午前でも午後でも周期的に出題されていますので，探索方法をしっかり押さえておきましょう。

関係なく**計算量は*O*(1)**となります。ただし，違うデータでハッシュ値が重なる**シノニム**という問題が発生することがあり，その場合には工夫が必要です。

それでは，次の問題を解いてみましょう。

問題

探索表の構成法を例とともにa～cに示す。探索の平均計算量が最も小さい探索手法の組合せはどれか。ここで，探索表のコードの空欄は表の空きを示す。

a コード順に格納した探索表

コード	データ
120380	……
120381	……
120520	……
140140	……

b コードの使用頻度順に格納した探索表

コード	データ
120381	……
140140	……
120520	……
120380	……

c コードから一意に決まる場所に格納した探索表

コード	データ
120381	……
120520	……
140140	……
120380	……

	a	b	c
ア	2分探索	線形探索	ハッシュ表探索
イ	2分探索	ハッシュ表探索	線形探索
ウ	線形探索	2分探索	ハッシュ表探索
エ	線形探索	ハッシュ表探索	2分探索

（平成22年秋 応用情報技術者試験 午前 問6）

解説

a～cの探索表をそれぞれ見ていきます。

a：コード順に格納した探索表の場合

線形探索では上から順に単純に見ていくため，計算量は*O*(n)になります。これに対し，**2分探索**を用いると，真ん中のデータと比較して半分にしていくので計算量が*O*(log n)とな

り，線形探索より効率的です。なお，ハッシュ値で格納されているわけではないので，ハッシュ表探索は使用できません。

b：コードの使用頻度順に格納した探索表の場合

線形探索で上から順番に見ていく場合，一番上に最も使用頻度の高いデータがあるので，通常の並びに比べて早く見つかる可能性が高くなります。なお，2分探索は整列されていないデータには使用できず，ハッシュ表探索はハッシュ値を使用していないデータには使用できません。

c：コードから一意に決まる場所に格納した探索表の場合

コードから一意に決まる場所はハッシュ値によって求められるので，**ハッシュ表探索**が計算量$O(1)$で使用できます。線形探索でも探索はできますが，データが格納されていない領域などもあるので効率は悪くなります。また，整列されていないデータなので2分探索は使用できません。

≪解答≫ア

整列アルゴリズム

整列のアルゴリズムは，昇順（小さい順）または降順（大きい順）にデータを並び替えるアルゴリズムです。代表的な整列アルゴリズムには以下の七つがあります。

①バブルソート

隣り合う要素を比較して，大小の順が逆であれば，その要素を入れ替える操作を繰り返すアルゴリズムです。隣同士を繰り返しすべて比較するので，**計算量は$O(n^2)$**となります。

②挿入ソート

整列された列に，新たに要素を一つずつ**適切な位置に挿入**する操作を繰り返すアルゴリズムです。挿入位置を決めるのに線形探索を行うため，**計算量は$O(n^2)$**となります。

勉強のコツ

整列のアルゴリズムはソフトウェア開発技術者試験ではよく出題されていましたが，最近はあまり出題されていません。
細かいところを完全に理解するより，それぞれのアルゴリズムの特徴や違い，計算量について押さえておくことが大切です。

発展

整列アルゴリズムは，①～③が基本3ソート，④～⑦が応用4ソートというかたちで分類されます。基本ソートの三つは，単純ですが時間がかかります。応用ソートは，それぞれアルゴリズムは複雑ですが，速度が改善されて効率良く整列を行うことができます。

③選択ソート

未整列の部分列から**最大値（または最小値）を検索**し，それを繰り返すことで整列させていくアルゴリズムです。最小値の探索を毎回行うため，**計算量は$O(n^2)$**となります。

④クイックソート

最初に**中間的な基準値**を決めて，それよりも大きな値を集めた部分列と小さな値を集めた部分列に要素を振り分けます。その後，それぞれの部分列の中で基準値を決めて，同様の操作を繰り返すアルゴリズムです。ランダムなデータの場合には，**計算量は$O(n \log n)$**となります。

⑤シェルソート

ある**一定間隔おきに取り出した要素**から成る部分列をそれぞれ整列させ，さらに間隔を狭めて同様の操作を繰り返し，最後に間隔を1にして完全に整列させるというアルゴリズムです。挿入ソートの発展形で，ざっくり整列させてから細かくしていくので効率が良くなります。間隔は，15，7，3，1……と，$2^n - 1$でnを一つずつ減らして狭めていくので，**計算量は$O(n \log n)$**となります。

⑥ヒープソート

未整列部分で**ヒープ**を構成し，その根から最大値（または最小値）を取り出して整列済の列に移すという操作を繰り返して，未整列部分をなくしていくアルゴリズムです。選択ソートの発展形であり，ヒープを使うことで，最大値（または最小値）を検索する作業を効率化しています。そのため，**計算量は$O(n \log n)$**となります。

⑦マージソート

未整列のデータ列を前半と後半に分ける分割操作を，これ以上分割できない，大きさが1の列になるところまで繰り返します。その後，**分割した前半と後半をマージ（併合）**して，整列済のデータ列を作成することを繰り返し，最終的に全体をマージするアルゴリズムです。**計算量は$O(n \log n)$**と効率的ですが，マージす

発展

クイックソートでは，ランダムなデータなどでうまく基準値を選べると非常に効率的な計算が行えるため，クイックソートの名のとおり，ランダムに並んだデータを整列させるときの速度は最も速くなります。しかし，最初から整列してあるデータなどではかえって非効率になり，最悪の場合，計算量が$O(n^2)$となってしまうことがあるため，使う場面には注意が必要です。

関連

ヒープについては，「1-2-1 データ構造」(P.60) を参照してください。

るための領域が必要となるので，作業領域（メモリ量）を多く消費することが欠点です。

それでは，次の問題を解いてみましょう。

問題

データの整列方法に関する記述のうち，適切なものはどれか。

ア　クイックソートでは，ある一定間隔おきに取り出した要素から成る部分列をそれぞれ整列し，更に間隔を詰めて同様の操作を行い，間隔が1になるまでこれを繰り返す。

イ　シェルソートでは，隣り合う要素を比較して，大小の順が逆であれば，それらの要素を入れ替えるという操作を繰り返す。

ウ　バブルソートでは，中間的な基準値を決めて，それよりも大きな値を集めた区分と小さな値を集めた区分に要素を振り分ける。次に，それぞれの区分の中で同様な処理を繰り返す。

エ　ヒープソートでは，未整列の部分を順序木に構成し，そこから最大値又は最小値を取り出して既整列の部分に移す。この操作を繰り返して，未整列部分を縮めていく。

（平成20年春 ソフトウェア開発技術者試験 午前 問11）

解説

ヒープソートでは，未整列の部分を順序木（ヒープ）に構成し，そこから最大値又は最小値を取り出していきます。したがってエが正解です。

ア　シェルソートの説明です。クイックソートは，ウの説明のとおり，中間的な基準値を決めるものです。

イ　バブルソートの説明です。シェルソートは，アの説明のとおり，一定間隔おきにデータを整列します。

ウ　クイックソートの説明です。バブルソートは，イの説明のとおり，隣り合う要素を比較して入れ替えます。

≪解答≫エ

再帰のアルゴリズム

発展
クイックソート，マージソートなどは，再帰を用いてプログラムを記述することもできます。

再帰とは，再び帰る，つまり，自分自身をもう一度呼び出すようなアルゴリズムです。関数などで，呼び出した関数自身を呼び出す場合が再帰に当たります。

言葉だけではイメージしづらいので，問題を見ながら再帰を感じてみましょう。

問題

次の数式は，ある細菌の第n世代の個数$f(n)$が1世代後にどのように変化するかを表現したものである。この漸化式の解釈として，1世代後の細菌の個数が，第n世代と比較してどのようになるかを説明しているものはどれか。

$$f(n+1)+0.2\times f(n)=2\times f(n)$$

ア　1世代後の個数は，第n世代の個数の1.8倍に増える。
イ　1世代後の個数は，第n世代の個数の2.2倍に増える。
ウ　1世代後の個数は，第n世代の個数の2倍になり，更に増殖後の20%が増える。
エ　1世代後の個数は，第n世代の個数の2倍になるが，増殖後の20%が死ぬ。

（平成21年春 応用情報技術者試験 午前 問5）

解説

この問題でいう，ある細菌の第n世代の個数を示す関数$f(n)$では，この値を計算するのに，同じ関数$f(n)$を，nの値を変えて再利用します。これが再帰です。

問題の式だと分かりにくいので，変形すると次のようになります。

$$f(n+1) = 1.8 \times f(n)$$

ここで，$f(n+1)$は$f(n)$の1.8倍になることが分かります。

そしてこの式から$f(n)$を求めるためには次の式を計算する必要があることが分かります。

$$f(n) = 1.8 \times f(n-1)$$

さらに，$f(n-1) = 1.8 \times f(n-2)$，$f(n-2) = 1.8 \times f(n-3)$……と，どんどんさかのぼっていく必要が出てきます。この計算自体が，再帰計算です。

ちなみに，問題自体は$f(n+1) = 1.8 \times f(n)$なので，単純に，1世代後の個数は第n世代の個数の1.8倍に増えます。

≪解答≫ア

文字列処理のアルゴリズム

文字列処理のアルゴリズムには，文字列の探索，置換などがあります。文字列の探索には，単純に前から探索する方法のほかに，BM法（ボイヤ・ムーア法）などの効率的な探索方法があります。置換は探索の後に行われるので，基本的に文字列探索と同じアルゴリズムです。

それでは，実際の午後問題を例に，文字列探索のアルゴリズムを学習していきましょう。

勉強のコツ

アルゴリズム問題は，実際のプログラムも大事ですが，その前の説明文をしっかり理解してプログラムと対応付けることが大切です。文章の部分もしっかり読みましょう。そして，その文章を理解することが，アルゴリズムの学習にもつながります。

問題

文字列照合処理に関する次の記述を読んで，設問１，２に答えよ。

ある文字列（テキスト）中で特定の文字列（パターン）が最初に一致する位置を求めることを文字列照合という。そのための方法として，次の二つを考える。ここで，テキストとパターンの長さは，どちらも１文字以上とする。

〔方法１　単純な照合方法〕

テキストを先頭から１文字ずつパターンと比較して，不一致の文字が現れたら，比較するテキストの位置（比較位置）を１文字分進める。この方法による処理例を図１に示す。

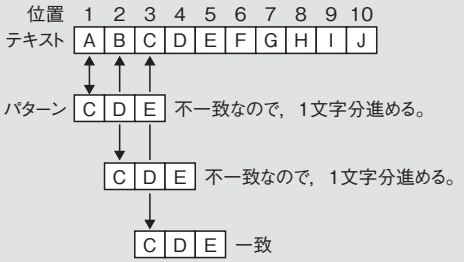


図１　単純な照合方法

この手順を基に，テキストとパターンを与えると，最初に一致した文字位置を返すアルゴリズムを作成した。このアルゴリズムを図２に示す。

なお，テキストは配列Ｔ，パターンは配列Ｐに格納されており，Ｔ[n]及びＰ[n]はそれぞれのｎ番目の文字を，T.length及びP.lengthはそれぞれの長さを示す。

```
function search1(T,P)
  for iを1から [　ア　] まで1ずつ増やす
    for jを1から [　イ　] まで1ずつ増やす
      if T[i+j−1]とP[j]が等しい                ←α
        if jと [　イ　] が等しい
          return i
        endif
      else
        break
      endif
    endfor
  endfor
  return −1    // パターンが見つからない場合は−1を返す
endfunction
```

図2　単純な照合方法のアルゴリズム

〔方法2　効率的な照合方法〕

方法1では，パターンとテキストの文字が不一致となった場合，比較位置を1文字分進めている。ここでは，比較位置をなるべく多くの文字数分進めることで，照合における比較回数を減らすことを考える。

パターンの末尾に対応する位置にあるテキストの文字（判定文字）に着目すると，次のように比較位置を進める文字数（スキップ数）が決定できる。

(1)　判定文字がパターンに含まれていない場合は，判定文字とパターン内の文字の比較は常に不一致となるので，パターンの文字数分だけ比較位置を進める。また，判定文字がパターンの末尾だけに含まれている場合も，同様にパターンの文字数分だけ比較位置を進める。

(2)　判定文字がパターンの末尾以外に含まれている場合は，判定文字と一致するパターン内の文字が，テキストの判定文字に対応する位置に来るように比較位置を進める。ただし，パターン内に判定文字と一致する文字が複数ある場合は，パターンの末尾から最も近く（ただし，末尾

を除く）にある判定文字と一致するパターン内の文字が，判定文字に対応する位置に来るように比較位置を進める。

この方法による処理例を図３に示す。

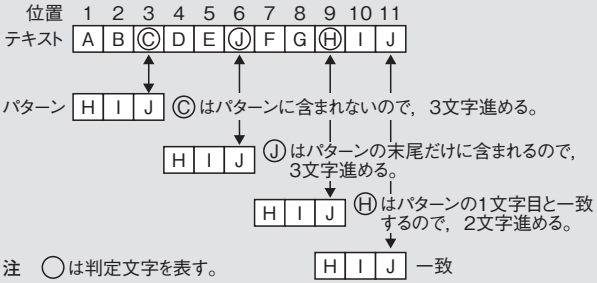


図３　効率的な照合方法

パターンが与えられると，判定文字に対応するスキップ数を一意に決定することができる。例えば，パターンHIJに対して判定文字とスキップ数の対応表を作成すると表１のようになる。

表１　パターンHIJに対するスキップ数

判定文字	H	I	J	その他の文字
スキップ数	2	1	3	3

この考え方を基に作成したアルゴリズムを図４に示す。

なお，テキストは配列T，パターンは配列P，スキップ数を決める表の判定文字は配列C，スキップ数は配列Dに格納されており，T[n]，P[n]，C[n]，D[n] はそれぞれのn番目の文字又は数値を，T.length及びP.lengthはテキストとパターンの長さを示す。

このアルゴリズムは，三つの主要部分から成っている。一つ目は，与えられたパターンについて判定文字とスキップ数の対応表を作成する処理である。ここでは，配列Cに格納される空白文字は表１の“その他の文字”及びパターンの末尾の文字“J”を表現するために使用している。テキストとパター

発展

さらに効率的な照合方法として，テキストとパターンを比較するとき，先頭からではなく最後の文字から順に比較する方法があります。この方法はBM法（ボイヤ・ムーア法）と呼ばれるもので，文字列探索では一般的に使われているアルゴリズムです。

ンは空白文字を含まないものとする。二つ目は，文字を比較する処理である。ここでは，現在の比較位置に対応したテキストとパターンを方法1と同様に1文字ずつ比較して，パターンに含まれる文字と対応するテキストの文字がすべて一致するか，不一致となる文字が見つかるまで繰り返す。三つ目は，判定文字とスキップ数の対応表を引いてスキップ数を決定する処理である。

```
function search2(T,P)
  for xを1からP.lengthまで1ずつ増やす
    C[x] ← " "                    // " "は空白文字を表す
    D[x] ← P.length
  endfor
// (1):表の作成
  for jを1からP.length−1まで1ずつ増やす
    for kを1からjまで1ずつ増やす
      if C[k]とP[j]が等しい or C[k]と"が等しい
        C[k] ← P[j]
        D[k] ← [  ウ  ]
        break
      endif
    endfor
  endfor
// 文字列を照合する
  i ← 1
  while iが [  ア  ] 以下である
// (2):文字の比較
    for jを1から [  イ  ] まで1ずつ増やす
      if T[i+j−1]とP[j]が等しい              ←β
        if jと [  イ  ] が等しい
          return i
        endif
      else
        break
      endif
    endfor
```

```
// (3):スキップ数の決定
    for kを1からP.lengthまで1ずつ増やす
      if C[k]とT[i+P.length−1]が等しい or C[k]と
      ”“が等しい
        d ← D[k]
        break
      endif
    endfor
    i ← i+d
  endwhile
  return −1
endfunction
```

図4　効率的な照合方法のアルゴリズム

設問1　図2及び図4の［ア］～［ウ］に入れる適切な字句を答えよ。

設問2　パターンHIPOPOTAMUSに対するスキップ数を表2に示す。［エ］，［オ］に入れる適切な数値を答えよ。

表2　パターンHIPOPOTAMUSに対するスキップ数

判定文字	H	I	P	O	T	A	M	U	S	その他の文字
スキップ数	エ	9	オ	5	4	3	2	1	11	11

（平成21年秋 応用情報技術者試験 午後 問2抜粋）

解説

設問1

・空欄ア，イ

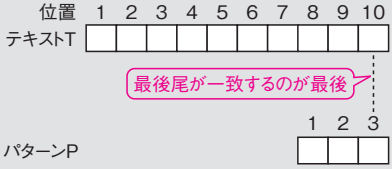
単純な照合方法のアルゴリズムについての穴埋め問題です。基本3構造，特にループ（繰返し）を中心に図2の構造を考えると，次のようになります。

```
function search1(T,P)
  for iを1から［　ア　］まで1ずつ増やす
    for jを1から［　イ　］まで1ずつ増やす
      if T[i+j−1]とP[j]が等しい
        if jと［　イ　］が等しい
          return i
        endif
      else
        break
      endif
    endfor                              ループ②
  endfor                                ループ①
  return −1    // パターンが見つからない場合は−1を返す
endfunction
```

方法の説明と，アルゴリズムを対応させていきます。

外側のループ①が，〔方法1　単純な照合方法〕の「テキストを先頭から1文字ずつパターンと比較して，不一致の文字が現れたら，比較するテキストの位置（比較位置）を1文字分進める」部分に該当します。つまり，先頭から1文字ずつ，最後まで比較していくループです。

このループが終わるのは，最後まで探索し終わったとき——といっても，テキストTが一致する可能性があるのは，パターンPの最後がパターンTの最後と重なるときまでです。そのため，以下のようなかたちになったときが最後になります。



最後尾が一致するときの位置は，「テキストTの長さ−パターンPの長さ」からは一つ進んだところになります。そのため，iは1から最後尾が一致するとき，つまり，**T.length −**

P.length＋1までとなります。そしてこれが，空欄アの答えです。

内側のループ2は，テキストTとパターンPを比較するループです。こちらは，パターンPを1文字ずつ最後まで比較していくので，ループの終わりはP.lengthになります。

・空欄ウ

効率的な照合方法のアルゴリズムについての穴埋め問題です。図4の効率的な照合方法のアルゴリズムでは，(1)：表の作成，(2)：文字の比較，(3)：スキップ数の決定，の三つの部分に分かれています。空欄ウは，(1)：表の作成の部分なので，その部分の構造を考えると以下のようになります。

```
// (1)：表の作成
for jを1からP.length−1まで1ずつ増やす        ループ③
  for kを1からjまで1ずつ増やす                ループ④
    if C[k]とP[j]が等しい or C[k]と”“が等しい
      C[k] ← P[j]
      D[k] ←  [  ウ  ]
      break
    endif                                    選択①
  endfor
endfor
```

このとき，ループ③とループ④で，パターンPと判定文字の配列Cを比較し，その結果，スキップ数の配列Dを求めます。このとき，判定文字がパターンの末尾以外に含まれている場合には，〔方法2　効率的な照合方法〕(2)に「判定文字と一致するパターン内の文字が，テキストの判定文字に対応する位置に来るように比較位置を進める」とあります。この位置がスキップ数になります。スキップ数は，最後尾の一つ前なら1，二つ前なら2……，先頭なら(パターンの文字数－1)となるので，空欄ウにはP.length－jが入ります。

設問2

先ほどの図4「(1)：表の作成」を使って，パターンHIPOPOTAMUSを例にトレースしてみます。このとき，P.length = 11です。j = 1，k = 1から順に考えていくと，最初のHでは，配列Cはまだ空白しか入っていないので，C［1］と""が等しくなり，C[1]←H，D[1]←11 − 1 = 10が入ります。同様に，HIPOまで終わらせると，配列CとDは次のようになります。

判定文字C	H	I	P	O	他
スキップ数D	10	9	8	7	11

このとき，次のPは3番目のPと一致します。その場合には，C[3]とP[5]が等しいことになるので，配列Dは上書きされて，D［3］←11 − 5 = 6となります。同様に繰り返していくと，最終的な表は以下のようになります。

判定文字C	H	I	P	O	T	…	他
スキップ数D	**10**	9	**6**	5	4	…	11

したがって，Hのスキップ数は**10**，Pのスキップ数は**6**です。

≪解答≫

設問1　ア：T.length − P.length + 1
イ：P.length，ウ：P.length − j
設問2　エ：10，オ：6

その他のアルゴリズム

その他のアルゴリズムとして定番のものには，グラフのアルゴリズム，近似・確率統計などの数学的なアルゴリズム，データ圧縮のアルゴリズム，図形描画のアルゴリズムなどがあります。

出題頻度は低いですが様々なものがありますので，見かけたらその仕組みを理解しておきましょう。

覚えよう！

- □ **基本3ソートの計算量は$O(n^2)$，応用4ソートの計算量は$O(n \log n)$**
- □ **プログラムの構造を追っていくときには，ループ（繰返し）がポイント**

コラム

アルゴリズム問題のポイント

アルゴリズム問題，特に午後問題で出される流れ図や疑似言語の問題を解くポイントとしては，次の三つが挙げられます。

1. プログラムの構造を明らかにする
2. 実際に値を入れて，トレースする
3. 最後の値を意識する

1の「プログラムの構造」とは，基本3構造のことです。特にループ（繰返し）を意識すると，プログラムの流れがよく見えてきます。そしてその構造を，実際の問題文でのアルゴリズムの説明と対比させることにより，プログラムの全体像が見えてきます。

2の「トレース」は，面倒がらずやることが大切です。実際に手を動かして，iの値を1，2……と増やしていきましょう。最後までやらなくても流れは見えてくると思いますが，最初からやらないとイメージが湧きません。

3の「最後の値」というのは，最後の値がnで終わるのかn－1で終わるのかといった，微妙な±1の範囲の話です。実は，プログラムのミスはこの±1の差で起こることが多いので，アルゴリズム問題ではここを聞いていることが多いのです。実際に問題を解くときには，ぜひ，この「最後の値」はていねいに導き出していきましょう。

最後に，アルゴリズム問題は「慣れ」が肝心です。実際にプログラムしたりアルゴリズム問題を解いたりして，解く感覚を磨いていきましょう。

1-2-3 プログラミング

プログラミングにおいては，いろいろなプログラムの構造やその動きを知ることが大切です。

プログラム構造

プログラムは，再使用できるか，同じものを呼び出すことができるかなどの観点で，次の四つの構造に分類されます。

①再使用可能（リユーザブル）プログラム

一度使用したプログラムを再度使用できるプログラムです。つまり，プログラムを一度終了させた後に再度起動させることが可能です。再使用が不可能なプログラムとは，一度終了させたら電源を切って再起動しないと動かないものなので，現在のプログラムはほとんどが再使用可能です。

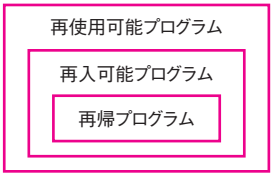
②再入可能（リエントラント）プログラム

プログラムが使用中でも，再度入ってもう一つ起動させることができるプログラムです。一つのプログラムを複数立ち上げることができ，その複数のプログラムを管理するために，共通の領域のほかに**それぞれのプログラムを独立で管理する領域**が必要になります。そして，再入可能プログラムは複数起動が可能なので，**必ず再使用可能プログラムでもあります。**

③再帰（リカーシブ）プログラム

自分自身を呼び出すことができるプログラムのことを，再帰プログラムと呼びます。再帰のアルゴリズムを使うようなプログラムで使用されます。そして，再帰プログラムは，自分自身に再入するので，**必ず再入可能プログラムでもあります。**

①～③の関係を図にすると，右のようになります。



各プログラムの関係

関連

逐次再使用可能プログラム（Serially Reusable Program）は，再使用可能プログラムの一種です。再使用可能プログラムのうち，複数の要求が来たときに一度に一つずつしか処理を行わないプログラムを指します。そのため，逐次再使用可能プログラムは再入可能プログラムとはなりません。

発展

再入可能プログラムの代表例としては，IE（インターネットエクスプローラ）などのWebブラウザがあります。一つのプログラムに対していくつも同じものを立ち上げることができます。そして，それぞれのWebブラウザは別のWebページを見に行くので，見に行ったWebページの情報を別々に格納しておく必要があります。

④再配置可能（リロケータブル）プログラム

プログラムをメモリ上で再配置することが可能なプログラムです。メモリのアドレスを指定する際，相対アドレスですべて指定しているプログラムは再配置可能になります。メモリの位置を指定する必要があるOSなどのプログラムのほかは，ほとんどの場合，再配置可能にすることができます。

それでは，実際の問題を解いてみましょう。

問題

再入可能（リエントラント）プログラムに関する記述のうち，適切なものはどれか。

ア　再入可能プログラムは，逐次再使用可能プログラムから呼び出すことはできない。
イ　再入可能プログラムは，呼出し元ごとに確保された記憶領域に局所変数が割り当てられる。
ウ　実行途中で待ち状態が発生するプログラムは，再入可能プログラムではない。
エ　逐次再使用可能なプログラムは，再入可能プログラムでもある。

（平成22年秋 応用情報技術者試験 午前 問8）

解説

再入可能プログラムは，**呼出し元ごとに確保された記憶領域に局所変数が割り当てられる**ので，選択肢イが正解です。再入可能プログラムの場合，複数のプログラムを区別して管理するために，呼出し元ごとに独自変数を管理する必要があります。

ア　再入可能プログラムは，ほかのプログラムから呼出し可能です。逐次再使用可能プログラムからでも呼び出すことができます。

ウ　待ち状態は，入出力の待ちなどで発生するので，プログラムの構造とは関係ありません。
エ　逐次再使用可能は再入可能プログラムではありません。また，再入可能プログラムは再使用可能です。

≪解答≫イ

覚えよう！

- □　再帰プログラムは再入可能プログラム
- □　再入可能プログラムは再使用可能プログラム

1-2-4 プログラム言語

プログラム言語は，人間と機械を結び付けるためにできたものです。コンピュータは機械語（マシン語）しか理解できないのですが，人間が機械語でプログラムを組もうとするととても大変です。そこで，機械語との橋渡しを行うために，いろいろなプログラム言語が登場してきました。

プログラム言語については，それぞれの特徴だけでなく，「なぜその言語ができたのか」という背景を知っておくと，より深く理解できます。

プログラム言語の変遷

プログラム言語は，人間の役に立つために，そして，時代の需要に合わせるために徐々に進化してきました。進化の流れを次表にまとめます。

進化の流れ →

種類	機械語	アセンブリ言語	高級言語	構造化言語	オブジェクト指向言語
特徴	0101011 110101	MOV AX ADD AY	Z＝X＋Y	if(a==b) while(1)	class A private b
言語例	—	CASLII, Z80	FORTRAN, COBOL	C，Pascal	Smalltalk, C++, Java

言語の種類

発展

基本情報技術者試験で出題されるアセンブリ言語（CASLII）は，仮想機械をイメージしたアセンブリ言語です。一度勉強してみると，機械がプログラムをどう処理していくのかイメージしやすくなると思います。

流れを追いつつ，それぞれの言語の特徴を見ていきましょう。

①機械語（マシン語）

コンピュータが解釈できるのは，2進数の機械語のみです。2進数の羅列なので人間にはとても分かりにくく，実際には16進数で表記させてプログラムしていました。

②アセンブリ言語

機械語と1対1で対応させることで書きやすくした言語です。加算演算をADD，データの移動をMOVなどで表現します。ただし，機械語を置き換えただけなので，人間の考え方とはだいぶ異なります。例えば，C＝A＋Bを計算する場合は次のようになります。

```
MOV  PA, A
MOV  PB, B
ADD  PA, PB
MOV  Z, PA
```

このように，一つ一つプログラムを考えるのが大変でした。

③高級言語（高水準言語）

人間にとってわかりやすい形式というコンセプトで書かれた言語です。高水準言語とも呼びます。前述のC＝A＋Bなどは，そのままC＝A＋Bと書けるようになりました。

人間にとって分かりやすくといっても，いろいろな手法があります。英語で文章を書くように事務処理を記述できる言語が**COBOL**です。また，数式で科学技術計算を行うための言語が**FORTRAN**です。

発展

一般にコンピュータの世界では，人間に近いものを高級（または高レベル），コンピュータに近いものを低級（または低レベル）と呼びます。高級言語とは，人間に近い言語という意味です。

④構造化言語

高級言語でプログラムするとき，適当に行をジャンプすることを繰り返していると，混沌として分かりづらいプログラム（これをスパゲティプログラムと呼びます）になりがちでした。それを解消しようと，ダイクストラという人が基本3構造（「1-2-2 アルゴリズム」P.62参照）を提案しました。プログラムを書くときには，順次，選択，繰返しの基本3構造だけで書き，適当にジャンプす

るためのGOTO命令は極力使わない「構造化プログラミング」という考え方です。

その結果登場したのが，構造化言語です。**C**や**Pascal**などがこれに該当します。選択を表すif文，繰返しを表すwhileやfor文が加わりました。さらに，関数やサブルーチンという，同じ処理をする単位にまとめるという考え方も，構造化言語あたりから進化してきました。

⑤オブジェクト指向言語

関数（及びサブルーチン）などによるプログラムでは，グローバル変数と呼ばれる，複数の関数で共通使用する変数の内容が，意図しない場所で書き換えられたりして不具合を起こすことが多くなりました。そこで，グローバル変数をなくし，共通で同じ変数を使用する関数をまとめる**クラス**という考え方が必要になりました。

また，クラス以外にも，オブジェクト指向というオブジェクトやクラスを中心とする考え方が提唱されました。それらはプログラミングに様々なメリットをもたらすため，新しい言語がいろいろ開発されています。最初にできたオブジェクト指向言語は**Smalltalk**で，その後，C言語の発展形である**C++**や**Java**言語などが登場しました。

Java言語では，CやC++で問題となっていた，使用が終わったメモリを解放しない**メモリリーク**という問題に対応するため，自動でメモリを解放する**ガーベジコレクション**という機能があります。

その他の言語分類

プログラム言語は，時代の流れ以外にも様々に分類できます。代表的なものを紹介します。

①手続型言語

処理手順を，1行1行順を追って記述する言語です。COBOL，FORTRAN，C，C++，Javaなど，通常のプログラム言語はほとんどがこれに該当します。

②関数型言語

関数の定義とその呼出しでプログラムを記述する言語です。再帰処理向きで，代表的なものにLispがあります。C言語の関数とは意味が異なる数学的な形式なので，注意が必要です。

③論理型言語

述語論理を基礎とした論理式でプログラムを記述する言語です。人工知能の研究開発で使用されるPrologなどが代表例です。

④スクリプト言語

プログラムを機械語にコンパイルするのではなく，スクリプト（台本）のように記述して1行1行処理する簡易的な言語です。UNIX上でシェルを動かすための**シェルスクリプト**や，Excelのマクロ言語などがあります。

Webブラウザ上で動くスクリプト言語として，Java言語と似た言語で記述するJavaScriptとJScriptがあります。これら二つは互換性が低いので，共通する部分をまとめて標準化したECMAScriptが作られました。Webサーバ上で動くJavaのスクリプト言語にはJSP（Java Server Pages）があります。なお，サーバ上では，スクリプト言語ではないJavaアプリケーションや**サーブレット**も動きます。その他，Webサーバ上で動くJava以外のスクリプト言語としては，Perl，Ruby，PHPなどがあります。

それでは，次の問題を解いてみましょう。

発展

簡易的といっても，最近はそのスクリプト言語で本格的なWebシステムを作る例も多くなってきました。PHPやPerl，Rubyなどは大規模な用途にも使われています。

用語

シェルとは，UNIX上でユーザからの指示を解釈し，プログラムの起動や制御を行うプログラムです。シェルスクリプトには，Bourne Shell（sh），C Shell（csh），TENEX C Shell（tcsh）など様々な種類があり，ユーザの好みによって置き換え可能です。

問題

HTMLだけでは実現できず，JavaScriptを使うことによってブラウザ側で実現可能になることはどれか。

ア　アプレットの使用　　イ　画像の表示
ウ　サーバへのデータの送信　　エ　入力データの検査

（平成22年春 応用情報技術者試験 午前 問7）

解 説

JavaScriptは，ブラウザ側にプログラムを送り，ブラウザ上で実行します。そのため，HTML文では実行できない**動的な処理**，例えば，入力されたデータが適正かどうか検査するなどの処理を実行できます。したがって，選択肢エの入力データの検査が正解です。

ア　アプレットは，HTML文の<applet>タグで使用できます。アプレットは，**ブラウザにダウンロード**して実行します。

イ　画像の表示は，HTML文の<img>タグで実現できます。

ウ　サーバへのデータ転送は，HTML文の<form>タグで囲った領域で<input>タグに入れられたデータを転送することで実現できます。

≪解答≫エ

共通言語基盤

共通言語基盤（CLI：Common Language Infrastructure）は，プログラム言語やコンピュータアーキテクチャに依存しない環境を定義したものです。様々な高級言語で書いたプログラムを，書き直すことなく他のプラットフォームでも使うことができます。マイクロソフトが策定した**.NET Framework**の基幹を構成する実行コードや実行環境などについての仕様です。

用語

.NET Frameworkとは，マイクロソフトが開発したアプリケーションの開発・実行環境です。.NET Frameworkでコンパイルしたプログラムは，LinuxやMac OSなど，Windows以外のOSでも動かすことができます。

覚えよう！

□　**サーバで動くJSP，サーブレット，クライアントで動くJavaScript，アプレット**

1-2-5 その他の言語

コンピュータ上でやりたいことを実現するために，プログラム言語以外の言語を用いることもあります。その代表的なものにマークアップ言語があります。

データ定義言語

データ定義言語（**DDL**：Data Definition Language）は，コンピュータのデータを定義するための言語や言語の要素です。XMLのデータを定義する**DTD**（Document Type Definition）や，データベース言語SQLの一部である**SQL-DDL**は，データ定義言語の一例です。

関連
SQL-DDLについては，「3-3-3　データ操作」(P.215)で詳しく取り上げます。

Webページを記述するためのマークアップ言語

Webページを記述するためのマークアップ言語には，次のようなものがあります。

①HTML（HyperText Markup Language）

Webページを作成するために開発された言語です。**ハイパーテキスト**と呼ばれる，通常のテキストのほかに他のページへのリンク（ハイパーリンク）を埋め込むことができるテキストを使用します。画像，音声，映像などのデータファイルもリンクで埋め込むことができます。

発展
HTML，XMLはともに，SGML（Standard Generalized Markup Language）から派生した言語です。XMLはより厳密に進化したのに対し，HTMLはあいまいさを残していましたが，XMLが一般化するにつれ，HTMLをXMLに適合させたいという需要がでてきたため，XHTMLができました。

②XHTML（Extensible HTML）

HTMLをXMLの文法で定義し直したものです。より厳密に記述のチェックを行います。例えば，XML文書であるため，次のようなXML宣言を行う必要があります。

```
<?xml version="1.0" encoding="Shift_JIS"?>
```

その他，要素名や属性名はすべて小文字でなければならない，必ず開始タグと終了タグで囲まれていなければならないなど，様々な制約があります。

過去問題をチェック
CSSについての問題が，応用情報技術者 平成22年秋 午後 問8に出題されています。また，HTMLについては平成22年春 午前 問7で，XMLについては平成21年秋 午前 問8で出題されています。
地味に少しずつ出題されていますし，実務でもよく使われているものなので，知識は押さえておきましょう。

③CSS（Cascading Style Sheet）

文章のスタイルを記述するためにできた言語で，HTMLや

XHTMLの要素をどのように表示するか定義します。文章の**構造と体裁を分離させる**という理念の下，文章の構造はHTMLで，体裁はCSSで記述します。

XML（Extensible Markup Language）

特定の用途に限らず，汎用的に使うことができる**拡張可能な**マークアップ言語です。文書構造の定義は，DTDで行います。

XML文書の正当性の水準には，次の二つがあります。

①整形式XML文書（well-formed XML Document）

XMLデータを記述するための文法に従ったXML文書です。DTDでの定義は存在しない，または定義に適合していなくても支障はありません。

②妥当なXML文書（valid XML Document）

DTDの定義に適合したXML文書です。整形式XML文書でもあります。

XMLでは，厳密に妥当なXML文書のみを許可することも，整形式の文書で手軽にXMLを利用することも可能です。

その他，XML文書を変換するために使われる**XSLT**（XSL Transformation），XML文書同士のリンクを設定するための**XLink**（XML Linking Language）など，XML文書を扱うための言語にも様々なものがあります。

それでは，問題を解いていきましょう。

問題

整形式（well-formed）のXML文書が妥当（valid）なXML文書である条件はどれか。

ア　DTDに適合している。
イ　XML宣言が完全に記述されている。
ウ　XMLデータを記述するための文法に従っている。
エ　エンティティ参照ができる。

（平成22年春 応用情報技術者試験 午前 問8）

解説

妥当（valid）なXML文書とは，DTDの定義に適合したXML文書です。したがって，選択肢アが正解です。

イ　XML宣言は，文法に従うための必要条件の一つです。
ウ　XMLデータを記述するための文法に従っているのは整形式のXML文書です。
エ　エンティティ参照とは，そのまま表記すると制御文字と間違われる文字をエスケープ文字に変換することです。例えば，＜は<，＞は&gt，＆は&などに変換します。これは定義済なので，XML文書ならいずれでも使用できます。

≪解答≫ア

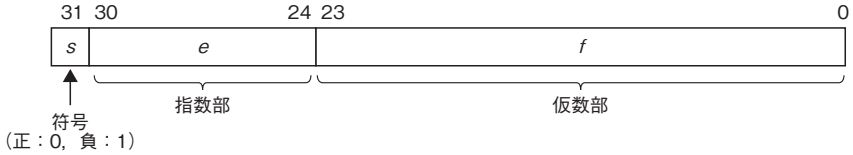
覚えよう！

□　**XML文書には，整形式XML文書と妥当なXML文書がある**

1-3 演習問題

問1 浮動小数点演算 CHECK ▶ □□□

次の浮動小数点表示法がある。小数点は仮数の左にあり，指数は64の“下駄履き表現”であって，値は$(-1)^s \times 0.f \times 2^{e-64}$である。二つの16進数45BF0000と41300000を，この浮動小数点表示法で表現された値として加算した結果は16進数でどう表現されるか。

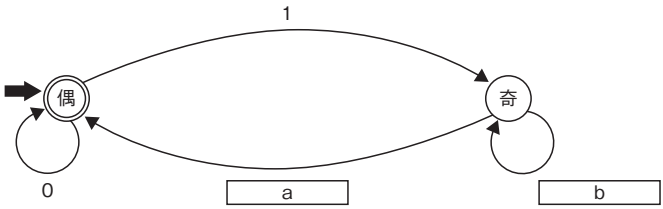


問2 逆ポーランド表記法 CHECK ▶ □□□

式A＋B×Cの逆ポーランド表記法による表現はどうなるか。

問3 オートマトン CHECK ▶ □□□

図は，偶数個の1を含むビット列を受理するオートマトンの状態遷移図であり，二重丸が受理状態を表す。a，bに入る数値は何か。



問4 サンプリング CHECK ▶ □□□

PCM伝送方式によって音声をサンプリング（標本化）して8ビットのディジタルデータに変換し，圧縮処理しないで転送したところ，転送速度は64,000ビット／秒であった。このときサンプリング間隔は何マイクロ秒か。

問5　関数の戻り値　CHECK ▶ □□□

文字列を引数とする関数len, first, butfirstを用いて, 関数compを再帰的に定義した。comp（"11", "101"）を呼び出したとき, 返される値は何か。

〔関数の定義〕

len(S)　：文字列Sの長さを返す。Sが空文字列のときは0を返す。

first(S)　：文字列Sの先頭の1文字のASCIIコードを返す。Sが空文字列のときはエラーを返す。

butfirst（S）：文字列Sの先頭の1文字を除いた残りの文字列を返す。Sが空文字列のときはエラーを返す。

```
comp(A, B)
begin
    if len(A) = 0 and len(B) = 0 then return   0;
    if len(A) = 0 and len(B) ≠ 0 then return   1;
    if len(A) ≠ 0 and len(B) = 0 then return  −1;
    if first(A) < first(B) then return   1;
    if first(A) > first(B) then return  −1;
    return comp(butfirst(A),butfirst(B));
end
```

問6　ハッシュ関数の衝突　CHECK ▶ □□□

キーが小文字のアルファベット1文字（a, b, …, zのいずれか）であるデータを, 大きさが10のハッシュ表に格納する。ハッシュ関数として, アルファベットのASCIIコードを10進表記法で表したときの1の位の数を用いることにする。衝突が起こるキーの組合せはどれか。ASCIIコードでは, 昇順に連続した2進数が, アルファベット順にコードとして割り当てられている。

ア　aとi　　イ　bとr　　ウ　cとl　　エ　dとx

問7 挿入ソート CHECK ▶ □□□

整列済みの列の末尾から比較して，次の要素の挿入位置を決める単純挿入整列法について考える。昇順に整列済みの大きさnのデータ列を，改めて昇順に整列する処理を行う場合の比較回数のオーダは何か。

問8 XML文書の変換 CHECK ▶ □□□

XML文書を，別の文書形式をもつXML文書やHTML文書などに変換するための仕様は何か。

1-4 演習問題の解答

問1　（平成20年春 ソフトウェア開発技術者試験 午前 問2改）

《解答》45C20000

まず，16進数45BF0000を値として表現します。

$(45BF0000)_{16} = (0\ 1000101\ 10111111000000000000000)_2$

上記から，指数部eは$(1000101)_2 = (69)_{10}$となります。そのため，値は次のようになります。

$(-1)^0 \times 0.10111111 \times 2^{69-64} = 0.10111111 \times 2^5 = (10111.111)_2$

次に，16進数41300000を値として表現します。

$(41300000)_{16} = (0\ 1000001\ 00110000000000000000000)_2$

上記から，指数部eは$(1000001)_2 = (65)_{10}$となります。そのため，値は次のようになります。

$(-1)^0 \times 0.0011 \times 2^{65-64} = 0.0011 \times 2^1 = (0.011)_2$

二つの値を加算します。

$(10111.111)_2 + (0.011)_2 = (11000.01)_2$

これを16進数で表現します。

$(11000.01)_2 = 0.1100001 \times 2^5 = (-1)_0 \times 0.1100001 \times 2^{69-64}$

指数部eは69，符号sは正(0)，仮数部fは1100001となります。これを以下のようにまとめます。

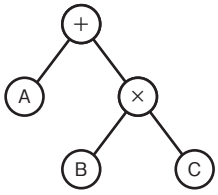
$(0100\ 0101\ 1100\ 0010\ 0000\ 0000\ 0000\ 0000)_2 = (\mathbf{45C20000})_{16}$

問2　（平成23年秋 応用情報技術者試験 午前 問2改）

《解答》ABC×＋

式A＋B×Cは中置表記法で表した式なので，木構造に置き換えると右図のようになります。

この木を逆ポーランド表記法（後置表記法）で，左部分木，右部分木，根の順に読んでいくと，**ABC×＋**となります。



問3 （平成19年春 ソフトウェア開発技術者試験 午前 問7改）

《解答》a：1，b：0

偶数個の1を含むビット列を受理するオートマトンの状態遷移図に，(偶)と(奇)という二つの状態があります。(偶)が受理状態なので，これが偶数個の1を含む状態を示していると推測でき，(奇)が奇数個の1を含む状態を示すことが分かります。ここで，ビット列なので0か1かのどちらかが来ますが，0が来たら1の数はそのまま奇数，1が来たら偶数になります。したがって，奇数から偶数への状態遷移である空欄aは**1**が来たとき，奇数のままでいる状態遷移である空欄bは**0**が来たときになります。

問4 （平成22年秋 応用情報技術者試験 午前 問3改）

《解答》125

8ビットのディジタルデータに変換したものを64,000ビット／秒の速度で転送したということは，64,000［ビット／秒］÷8［ビット］＝8,000［回／秒］のサンプリングを行ったことになります。そのため，サンプリング間隔をマイクロ秒（$=10^{-6}$秒）の単位で考えると，以下のようになります。

1,000,000［マイクロ秒／秒］÷8,000［回／秒］＝**125**［マイクロ秒］

問5 （平成21年春 応用情報技術者試験 午前 問7改）

《解答》−1

実際に，再帰的に値をトレースして値を求めます。

①1度目のcomp（"11"，"101"）呼出し

A＝"11"，B＝"101"なので，次のようになります。

len(A)＝2，len(B)＝3，first(A)＝"1"，first(B)＝"1"

五つのif文の条件には当てはまらないので，butfirst(A)＝"1"，butfirst(B)＝"01"で，関数compを再帰呼出しします。

②2度目のcomp（"1"，"01"）呼出し

A＝"1"，B="01"なので，次のようになります。

len(A)＝1，len(B)＝2，first(A)＝"1"，first(B)＝"0"

五つ目のif文の条件，first(A)＞first(B)に当てはまるので，return −1で−1を返し，①の関数compに戻ります。

return comp(butfirst(A)，butfirst(B))なので，戻ってきた値をそのまま返します。したがって，最終的な戻り値は**−1**となります。

問6

（平成23年特別 応用情報技術者試験 午前 問8）

《解答》エ

ASCIIコードは，アルファベット順に連続して並んでいます。10進数で表すと，最初のa（小文字）は97，次のbが98……と順に連続しています。そのため，アルファベット全文字を，10進表記法で表したときの1の位でハッシュ値をとると，右の表のようになります。

選択肢の組合せでは，**エ**のd（100）とx（120）がともに1の位が0なので，衝突が起こります。

1の位の数	アルファベット			
0		d	n	x
1		e	o	y
2		f	p	z
3		g	q	
4		h	r	
5		i	s	
6		j	t	
7	a	k	u	
8	b	l	v	
9	c	m	w	

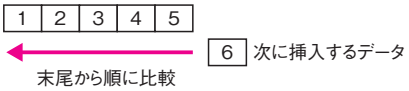
問7

（平成20年秋 ソフトウェア開発技術者試験 午前 問11改）

《解答》*O*(n)

単純挿入整列法（挿入ソート）の比較回数のオーダは，通常の場合，$O(n^2)$ です。しかし，問題文のようにすでに昇順に整列済みの場合は条件が異なります。

例えば，途中の段階で以下のような整列済みデータ列があるとします。



ここで，最初の5と比較した時点で5＜6となるので，次の6の挿入位置は末尾に確定します。最初から整列しているデータに対して順に大きい値を挿入する場合には，各データについて1回比較すればよいことになります。つまり，大きさnのデータ列での比較回数はn－1回（2番目のデータから挿入のための比較を開始するため）となり，オーダとしては，***O*(n)** となります。

問8

（平成21年秋 応用情報技術者試験 午前 問8改）

《解答》XSLT

XML文書を，別の文書形式をもつXML文書やHTML文書などに変換するための仕様を**XSLT**（XML Stylesheet Language Transformation），XSL変換といいます。XML文書全体または一部分について変換を行い，別のXML文書，またはHTMLなどの別の表現形式，TeXなどの印刷用形式の文書を生成することができます。