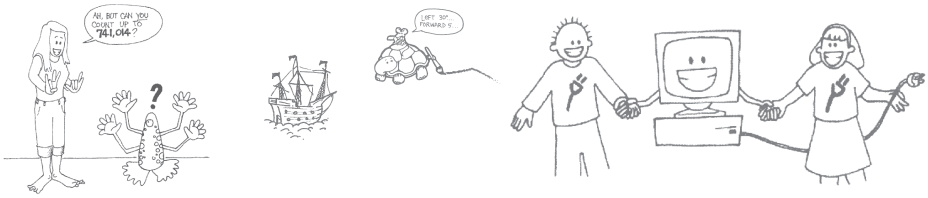


コンピュータを使わない情報教育
アンプラグドコンピュータサイエンス

COMPUTER SCIENCE Unplugged

Created by Tim Bell, Ian H. Witten and Mike Fellows

兼宗 進[監訳]

イーテキスト研究所

本書の複写利用は学校における教師自身による授業に限定されます。
© 2005 Computer Science Unplugged（www.unplugged.canterbury.ac.nz）

はじめに

現代ではあらゆる場所でコンピュータが使われています。私たちはそれらの使い方を学び、毎日使っています。しかし、コンピュータはどのように動いているのでしょう？どうやって考えているのでしょう？より速く、より良く使っていくにはどうしたらよいのでしょう？コンピュータ科学はこれらの問いに答える魅力的な題材です。この本の面白くて役に立つところは、コンピュータがどのように動くかを、ブロックを組み立てるように子ども向けに書いてあることです。それもコンピュータをまったく使わずに！

この本は課外活動や正課の授業で活用できます。コンピュータの専門家でなくても、子どもと楽しく学べます。この本はコンピュータの内部の仕組みをわかりやすく解説した「学習」で構成されています。問題には解答が用意されており、各学習の最後では「実際のコンピュータでは」という節で学習内容をまとめています。

多くの学習は数学を基礎にしています。たとえば２進数では、写像、グラフ、パターン、並べ替え、暗号化を学べます。その他、技術教育とコンピュータの動作原理にも関連しています。子どもは意味のある題材を使って、通信、問題解決、創造性、考え方を体験していきます。

この本は３つのコンピュータ科学の講義と、２人の学校教員と、教室での体験をもとに書かれました。私たちの体験では、コンピュータを使わずに、たくさんの概念を教えることが可能ということがわかりました。実際、コンピュータは学習の妨げになることもあります。コンピュータを消して、コンピュータ科学の本質について学びましょう！

謝辞

たくさんの子どもと先生が協力してくれました。ビクトリア（カナダ ブリティッシュコロンビア州）のサウスパーク校、クライストチャーチ（ニュージーランド）のシャーリー小学校、イラーム小学校、ウェストバーン小学校では子どもと先生が多くの学習活動で実験に協力してくれました。特に、我々を授業に参加させていただき、この学習活動の改善のための有益な助言をいただいた Linda Picciotto 、Karen Able 、Bryon Porteous 、Paul Cathro 、Tracy Harrold 、Simone Tanoa 、Lorraine Woodfield 、Lynn Atkinson に感謝いたします。 Gwenda Bensemann にはいくつかの学習活動を試して改善すべき点を提案していただきました。 Richard Lynders と Sumant Murugesh は授業での実践を助けていてだきました。暗号の学習活動は Ken Noblitz に開発していただきました。いくつかの学習活動は Kathy Beveridge の協力のもと、ビクトリア Mathmania グループに開発していただきました。初期の版の挿絵は Malcolm Robinson と Gail Williams に描いていただき、Hans Knutson にもそれらについてのアドバイスをいただきました。 Matt Powell には Unplugged プロジェクトの開発全般にわたって貴重な援助をしていただきました。この本を作るための初期の段階で多大な援助をいただいた、Brian Mason Scientific and Technical Trust に感謝いたします。

多くの活動を試していただき、数多くの有益な提案をしていただいた Paul と Ruth Ellen Howard に特に感謝いたします。 Peter Henderson 、Bruce McKenzie 、Joan Mitchell 、Nancy Walker-Mitchell 、Gwen Stark 、Tony Smith 、Tim A. H. Bell 、Mike Hallett 、Harold Thimbleby たちからもたくさんのコメントをいただきました。

我々は家族にも助けられました。 Bruce 、Fran 、Grant 、Judith 、Pam には活動を支えてもらいました。 Andrew 、Anna 、Hannah 、Max 、Michael 、Nikki はこの仕事の発想の源であり、多くの活動を最初に試してもらう存在でした。 Michael は文の圧縮の学習活動も考えてくれました。

Google は Unplugged プロジェクトと、この版の無料ダウンロードを支援してくれました。

この本で紹介した学習活動へのコメントや助言をお待ちしております。
http://www.unplugged.canterbury.ac.nz からご連絡ください。

日本語版発刊にあたって

21 世紀の情報社会を迎え、人々はコンピュータを使いながら生活するようになりました。これからの社会を生きていく子どもたちには、単に「コンピュータを使える」というだけでなく、「コンピュータを使いこなして生きていく」力が求められています。

コンピュータの進歩は速く、使い方を覚えてもすぐに陳腐化してしまいます。一方、コンピュータの基本的な原理はすぐには変わらないため、いちど理解してしまえば長い間活用していくことができます。「時代が変わっても通用する力」こそが、学校教育で伝えるべき知識ではないでしょうか。

ニュージーランドのティム・ベル博士たちが書いた『Computer Science Unplugged』は、学ぶのが難しいと思われていたコンピュータの基本原理を、小学生からわかりやすく学ぶことができる新しいメソッドです。

原著者たちは普段、コンピュータアルゴリズムの専門家として数式に囲まれながら研究を進めているはずですが、この本では 10 年以上前に、ティム・ベル博士が当時小学生だったお嬢さんに教えたときの体験を元に書かれているため、とても楽しく、わかりやすい内容になっています。

その後、この教具を使った学習法は多くの学校で改良を重ね、各国で公開されてきました。このたび、日本でもこの効果的な学習法を出版することができ、とても光栄に感じています。国内で行った実験授業では、小学生でも理解できるというだけでなく、特に中学校と高校の技術や情報の授業で大きな効果があることがわかりました。

今後は、本書の内容を補足・発展する形で、次のような活動を進めて行きたいと考えています。

（1）本書の教員向け補足ページ

各章の日本の教育課程への対応や、実際に学校の授業で使うための指導案など、授業で活用できる情報を、Web で積極的に公開していきたいと考えています。

アンプラグドの Web ページ : http://www.etext.jp

（2）アンプラグドメソッドの応用

アンプラグドの考え方は、本書で扱われている内容以外にも応用可能です。日本の教育課程に合わせて、新しいアンプラグド教材を開発する試みも行って行きたいと考えています。

（3）教育用プログラミング言語への発展

「楽しくコンピュータを学ぶ」上で、教具を使うアンプラグドメソッドでコンピュータの基本原理を体験した後は、「楽しくプログラミングを学ぶ」ことに挑戦してみてください。訳者の私たちは、初心者の子どもや先生が楽しくプログラムを作れるようになる「ドリトル」というプログラミング環境を研究しています。パソコンにインストールしなくても Web ブラウザで動かすことができますので、ぜひアンプラグドの後で授業で試してみてください。小学校から高校、大学までの授業テキストも用意されています。

ドリトルの Web ページ　http://dolittle.eplang.jp

この日本語版の作成にあたっては、多くの方々のお世話になりました。翻訳は、国士舘大学文学部の正田良先生、愛知教育大学教育学部の鎌田敏之先生、静岡大学教育学部の紅林秀治先生と共に進めました。各章の補足は筑波大学の久野靖先生に追補として解説していただきました。本書の内容は、松阪市立飯南中学校の井戸坂幸男先生と神奈川県立松陽高等学校の保福やよい先生に、教材作成を含めて授業で検証していただきました。

また、ドリトル教育メーリングリストのみなさま、教育マイクロワールド研究会および教育プログラミング研究会のみなさま、そして三重大学教育学部の奥村晴彦先生、大阪学院大学の西田知博先生をはじめとする情報処理学会コンピュータと教育研究会および初等中等情報処理教育委員会のみなさまに、多くの示唆をいただきました。

韓国ではソウル市の高麗大学を中心にアンプラグドの研究が進められており、李元揆教授と、その研究室に所属する崔淑敬さん、青木浩幸さんたちのメンバーから、多くの情報を提供していただきました。

2007 年 7 月
兼宗　進

目次

はじめに Ⅰ
謝辞 Ⅱ
日本語版発刊にあたって Ⅲ
第1部　データ：情報を表す素材 ---------- 1
学習１　点を数える　（２進数） 3
学習２　色を数で表す　（画像表現） 14
学習３　それ、さっきも言った！　（テキスト圧縮） 23
学習４　カード交換の手品　（エラー検出とエラー訂正） 31
学習５　２０の扉　（情報理論） 37
第2部　コンピュータを働かせる：アルゴリズム----------43
学習６　戦艦　（探索アルゴリズム） 45
学習７　いちばん軽いといちばん重い　（整列アルゴリズム） 64
学習８　時間内に仕事を終えろ　（並び替えネットワーク） 71
学習９　マッディ市プロジェクト　（最小全域木） 76
学習10　みかんゲーム　（ネットワークにおけるルーティングとデッドロック） 81
第3部　コンピュータに何をすべきか教える：手続きの表現----------85
学習11　宝探し　（有限状態オートマトン） 87
学習12　出発進行　（プログラミング言語） 102

APPENDIX　追補
学習１　点を数える（２進数）…… 108
学習２　色を数で表す（画像表現）…… 108
学習３　それ、さっきも言った！（テキスト圧縮）…… 108
学習４　カード交換の手品（エラー検出とエラー訂正）…… 109
学習５　２０の扉（情報理論）…… 110
学習６　戦艦（探索アルゴリズム）…… 110
学習７　いちばん軽いといちばん重い（整列アルゴリズム）…… 111
学習８　時間内に仕事を終えろ（並び替えネットワーク）…… 112
学習９　マッディ市プロジェクト（最小全域木）…… 114
学習１０　みかんゲーム（ネットワークにおけるルーティングとデッドロック） 114
学習１１　宝探し（有限状態オートマトン）…… 117
学習１２　出発進行（プログラミング言語）…… 118

第1部
データ

情報を表す素材

データ：情報を表す素材

コンピュータの中に情報をどうやって入れるか？

コンピュータ（computer）という言葉は、計算や２つの数の足し算を示すラテン語の computare から来ています。しかし、現在のコンピュータは計算が速いだけではありません。コンピュータは図書館にもなりますし、文章を書いたり情報を探したり、音楽を演奏したり、映画を見るためにも使えます。これらの情報はどのようにコンピュータに入っているのでしょう？信じられないかもしれませんが、実はコンピュータは０と１という２つのものしか扱えないのです！

データと情報はどう違うか？

データはコンピュータが直接処理できる数字です。コンピュータは人間が理解できるように、それらのデータを単語、数値、画像などの情報に変換します。

数値、文字、単語、画像などは、どのように０と１に変換されるか？

この章では、次のことを学びます。２進数、コンピュータの絵の描き方、ファクシミリの動作原理、データの効率のよい格納法、伝送エラーを防ぐ方法、格納する情報量の計り方。

学習１　点を数える

２進数

概要

コンピュータの中のデータは０と１の列の形で格納され、転送されます。たった２つの記号を使って、どうやったら文字や数字を表現できるのでしょう？

教科学習との関連

・Mathematics: Number Level 2 and up. Exploring numbers in other bases. Representing numbers in base two.

※［数と計算］２進法や他の記数法で数を表すこと

・Mathematics: Algebra Level 2 and up. Continue a sequential pattern, and describe a rule for this pattern. Patterns and relationships in powers of two.

※［数列］はじめの数列の何項かをみて続きを書いたり、数列を生成している規則を記述したりすること。特に公比２の等比数列でのパターンと関係。

技能

・Counting 数を数える。

・Matching 他の情報の表しかたに対応させる。

・Sequencing 順番に並べる。

年齢

・７歳以上

教材

・説明のために、５種類の２進数カード（６ページを参照）を作る必要があります。点の代りにスマイルマークのシールが貼ってある A4 のカードも効果的です。

＜子どもごとに必要なもの＞

・５種類のカード

・２進数が印刷されたカード（６ページ）をコピーして、切り取ってください。

・ワークシート学習：２進数（５ページ）

＜必要に応じてさらに学習を選ぶことができます＞

・ワークシート学習：２進数を計算する（７ページ）

・ワークシート学習：秘密のメッセージを送ろう（８ページ）

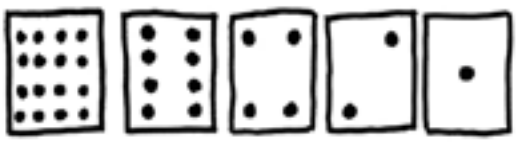
・ワークシート学習：電子メールとモデム（９ページ）

・ワークシート学習：31 より大きい数を数える（10 ページ）

・ワークシート学習：２進数のあれこれ（11 ページ）

基本的原理

5 ページのワークシートに進む前に、基本的な原理を説明しておきます。この学習では、ここに示す5種類のカードが必要です。片面だけに点が描かれています。教室の前でカードを持ってもらうために、5 人の子どもを選びましょう。カードは次の順番に並べる必要があります。



討論

カードの点の数に気づいたでしょうか（それぞれのカードは、右のカードの倍の数になっています）。 左にもう一枚加えるとしたら、点はいくつ必要ですか（32）。その次は？

このカードをめくることで、数を作ることができます。子どもに 6 を作らせてみてください（4 と 2 のカード)。次に 15（8 と 4 と 2 と 1 のカード)、次に 21（16 と 4 と 1)、...

次に、0 から順に数え上げさせてみましょう。

見ている子どもたちにカードがどのようなパターンで反転するかを観察させます（それぞれのカードは、その右のカードの半分の頻度で反転します)。 2 つ以上のグループで試してもよいでしょう。

カードは、表が見えていないときは 0 を表し、見えているときは 1 を表します。これが2進法の仕組みです。

子どもに 01001 を作らせます。 10進数だと何の数になりますか（9）。 17 は2進数でどう書きますか（10001）。子どもが理解するまで、いくつかの課題を与えます。

理解を充実するために、5 個の選択学習が用意されています。子どもが理解できるまで行いましょう。

解説シート

２進数

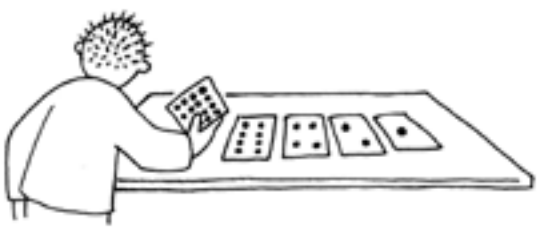
数え方を学ぼう

数え方は知っている？そうですね、ここでは新しいやり方を学びましょう！

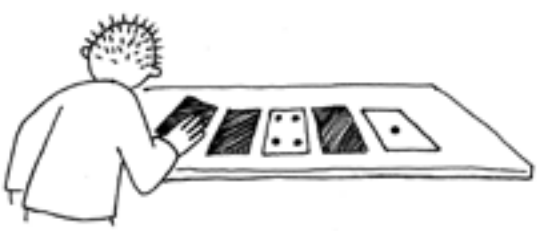
コンピュータは 0 と 1 しか使えないことを知っていましたか？ コンピュータで見えるものや聞こえるもの、たとえば文字、絵、数、動画、音などは、すべて 0 と 1 だけで表現されています。今回の学習を体験すれば、コンピュータと同じやり方で、友達に秘密のメッセージが送れるようになります。

説明

カードをシートから切りぬいて、16 個の点のカードを左にして図のように順に並べます。カードはきちんと図の順番に並べるように気をつけましょう。



次に、カードを裏返して、合わせて 5 個の点だけが見えるようにします。カードの順番



は変えてはいけません！

3 と 12 と 19 を作ってみよう。

ある数を表すのに、２通り以上の並べ方はありますか？

これらを使って表せるいちばん大きい数は何ですか？

いちばん小さな数は？

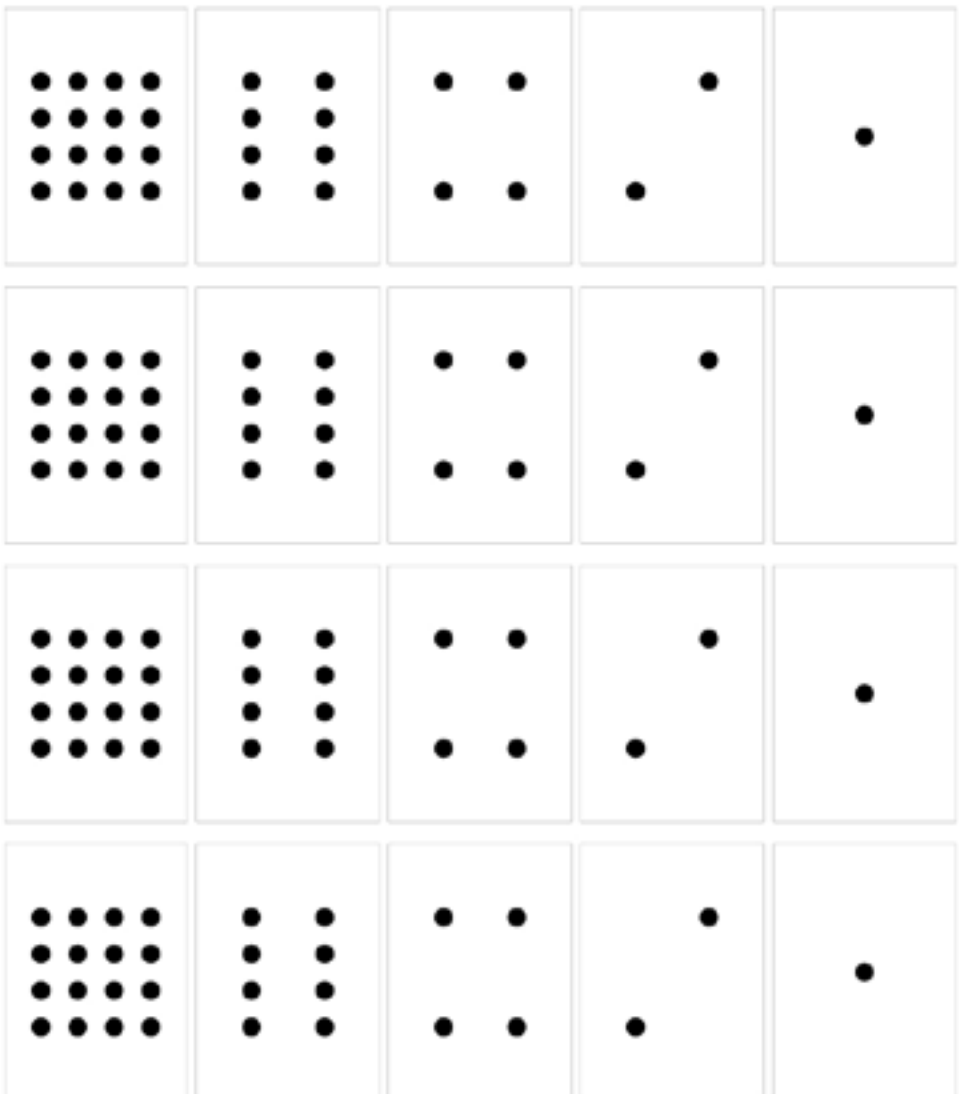
いちばん小さい数といちばん大きい数の間で、表せない数はありますか？

応用問題

1 、2 、3 、4 の数を順に作ってみよう。

数を 1 ずつ増やしたときに、カードをどのように裏返せばよいかという規則を見つけられますか？

複写用シート



ワークシート

２進数を計算する

２進法では、カードが向いている面を０と１で表します。０はカードが裏向きで、1 はカードが表向きで数が見える状態です。例を見ましょう。

10101 を計算できますか？ 11111 はどうでしょう？

自分の誕生月を２進数で書いてみましょう。次に、友だちの誕生日を２進数で書いてみましょう。

記号で書かれた数を計算しよう

☒☑☒☒☑ =
(☑=1, ☒=0)

⇧⇩⇧ =
(⇧=1, ⇩=0)

○○○○○ =
(⊙=1, ○=0)

=
(=1, =0)

☹ =
(☺=1, ☹=0)

=
(=1, =0)

=
(=1, =0)

+ + × + =
(+=1, ×=0)

↻↻↻↻↻ =
(↻=1, ↺=0)

▲▼▲▼▼ =
(▲=1, ▼=0)

♠♠♠♠♠ =
(♠=1, ♣=0)

応用問題

長さが 1 、2 、4 、8 、16 の棒を使って、31 までの長さの棒を作ってみよう。

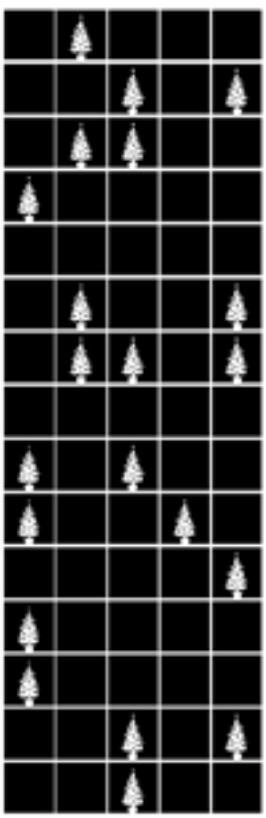
ワークシート

秘密のメッセージを送ろう

トムはデパートの最上階に閉じ込められてしまった。季節はそろそろクリスマスで家に帰りたい。どうしたらいいだろう？

声を出して叫んでみたが、誰も近くにいないようだ。道の向かい側では、夜遅くまでコンピュータの仕事をしている人たちが見えた。彼らに気づいてもらう方法はあるだろうか。トムは使えそうなものがないか探してみた。そして、アイデアが浮かんだ。クリスマスツリーの明かりでメッセージを送ろう！

彼はツリーの明かりを点けたり消したりできるように、コンセントを探した。そして道の向こうの女性が理解してくれそうな、簡単な２進数のコード（符号）を使った。うまく伝わるだろうか？

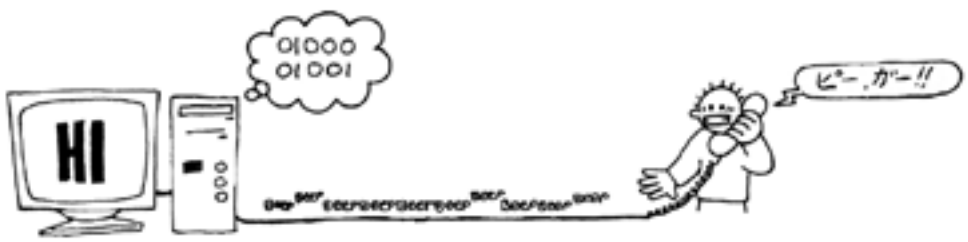


1	2	3	4	5	6	7	8	9	10	11	12	13
a	b	c	d	e	f	g	h	i	j	k	l	m
14	15	16	17	18	19	20	21	22	23	24	25	26
n	o	p	q	r	s	t	u	v	w	x	y	z

ワークシート

電子メールとモデム

コンピュータはモデムでインターネットに接続し、同じように２進法を使ってメッセージを送ることができます。違いは「ビー」という音を使うことだけです。高い音は１を表し、低い音は０を表します。これらの音はとても速く繰り返されるので、私たちに聴こえるのは耳障りな高い音です。もし聴いたことがなければ、インターネットに接続されたモデムの音を聴いてみましょう。または、ファクシミリに電話をかけましょう。ファクシミリも情報を送るためにモデムを使っています。



トムがデパートで使ったコード（符号）を使って、友だちに電子メールのメッセージを送ってみよう。ゆっくりで構いません。本物のモデムのような速さである必要はありませんから。

ワークシート

31 より大きい数を数える

もういちど 2 進数のカードを見てみましょう。もう 1 枚カードを加えるとしたら、何個の点のカードが必要でしょう？その次のカードは？新しいカードの点はどんな規則でしょう？これから見るように、大きな数も数枚のカードで表すことができます。

カードの並びをよく見ると、おもしろい規則がわかるでしょう。

1, 2, 4, 8, 16…

1+2+4 の答えは何でしょう？

次に 1+2+4+8 の答えは？

先頭から数を足していくとどうなるでしょう？

みなさんは指で数えることができますが、10 より大きい数は数えられませんよね。エイリアンではないのですから。でも、2 進法を使って、片手の指を点の書かれたカードとして使えば、みなさんは 0 から 31 の数を表すことができます。 32 個の数ですね。（ 0 も数です！）

指を使って順に数えてみましょう。指が伸びていたら 1 、閉じていたら 0 で す。

両手を使うと、実は 0 から 1023 の数を数えられます。 1024 個の数ですね！

もし足の指まで使えたら（宇宙人になる必要があるかもしれませんね）、もっと大きな数を数えられるでしょう。片手の指で 32 個を数えられれば、両手では 32 × 32 = 1024 個を数えられます。さて、足の指を自由に曲げられる女の子は、両手と両足を使っていくつの数まで数えられるで`　こ^

ワークシート

２進数のあれこれ

① ２進数のおもしろい性質を見ましょう。右側に０を加えるとどうなりますか。普段使っている１０進数では、右に０を加えると、元の数の 10 倍になりました。たとえば、9 は 90 になるし、30 は 300 になります。
では、２進数で右に０を加えるとどうなりますか？試してみてください。

1001 → 10010
（9）　（？）

他の数で、あなたの考えを試してみましょう。どんな規則がありますか？どうしてそうなると思いますか？

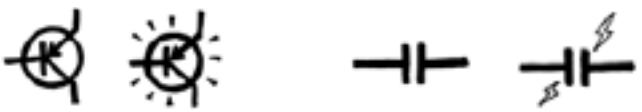
② 私たちが使ったカードは、コンピュータの「ビット」を表していました。ビットというのは２進数の１桁です。英語で使うアルファベットは、5 枚のカード、または 5 ビットがあれば表すことができます。しかし、コンピュータは大文字、小文字、数字、句読点、「$」や「~」のような記号などを区別できる必要があります。

キーボードを見ながら、コンピュータは英語のいくつの文字を表現できる必要があるのかを考えてみましょう。それらの文字を表現するために、コンピュータは何ビットが必要でしょう？

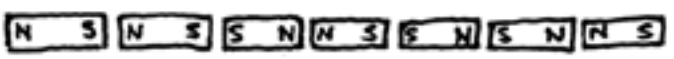
多くのコンピュータは、文字をそれぞれのビットで表現する ASCII と呼ばれる形で表現します。しかし、英語以外の国では、もっと長いコードを使う必要があります。

実際のコンピュータでは

現在のコンピュータは、情報を２進法で扱っています。２個の数字を使うので２進と呼ばれます。人間は普通、10 進を使います。１桁の０か１をビットと呼びます。ビットはコンピュータのメモリの中で、トランジスタがオンかオフか、またはコンデンサが充電されているかどうかで表現されます。



データが電話線や電波で送信されるときは、高い音と低い音が１と０として使われます。フロッピーディスクやハードディスクなどの磁気ディスクや磁気テープでは、ビットは表面に塗られた磁性体の N と S の向きで表されます。



音楽 CD や CD-ROM 、DVD では、表面で光が反射するかしないかによってビットを表します。1 個ずつのビットではたくさんの情報を表現できないため、普通は８個をひとまとまりにして 0 から 255 の値を表現します。この８ビットをバイトと呼びます。

コンピュータの速さは、いちどに計算できるビット数で変ります。たとえば、32 ビットのコンピュータはいちどに 32 ビットを計算します。 16 ビットのコンピュータは 32 ビットの数をいくつかに分けて計算する必要があるため遅くなります。

つまり、ビットやバイトは、コンピュータが記憶したり通信したりするための数値や文章、そしてすべての情報として使われます。この後の学習では、コンピュータで使われるその他の情報の表現についても見ていくことにします。

解答とヒント

2進数（5 ページ）

3 は 1 と 2 のカードが必要です。
12 は 8 と 4 のカードが必要です。
19 は 16 と 2 と 1 のカードが必要です。

数を作る方法は 1 通りしかありません。

作ることができるいちばん大きい数は 31 です。いちばん小さい数は 0 です。その間の数はすべて作れます。数を作る方法は 1 通りしかありません。

応用問題

どんな数でも、1 増やすためには、右から順に 1 枚を表にするまで裏返していきます。

2 進数で解く（7 ページ）

10101 = 21
11111 =31

秘密のメッセージを送る（8 ページ）

コード（符号）になったメッセージ : help i (') m trapped

31 より大きい数を数える（10 ページ）

いちばん小さい桁から足していくと、ある桁までの和は、その次の桁の数より 1 だけ小さい数になります。たとえば、1 ＋ 2 ＋ 4 は 7 で、次の 8 より 1 だけ小さいです。

足の指を自由に曲げられる女の子は、1024 × 1024 = 1,048,576 通り（0 から 1,048,575 まで！）を数えられます。

2進数のあれこれ（11 ページ）

2進数の右に 0 を置くと、その数は 2 倍になります。

すべての桁の 1 が 2 倍の価値を持つようになるため、全体で 2 倍の値になります。（10 進数では、右に 0 を置くと 10 倍になります）

コンピュータは英語のすべての文字を表すのに 7 ビット必要です。 7 ビットは 128 文字を表せます。7 ビットの文字は 8 ビットである 1 バイトの中に入れられるので、アルファベットしか使わない場合は 1 ビットが無駄になっていることになります。

学習 2　色を数で表す

画像表現

概要

コンピュータは絵や写真などを数字で記録します。この章ではどのように記録されるかを見て行きます。

教科学習との関連

・Mathematics: Geometry Level 2 and up. Exploring Shape and Space.

※［図形］いろいろな図形。

［数と計算］座標の考え方は小学校での学習指導要領には対応しませんが、小学校 1 年での「何番目」の発展として考えることもできます。しかし 10 を超える足し算が必要になるので、小 2 以降が無難かもしれません。

※［数量関係（中）］座標は中 1 で学びます。しかし、このような扱いなら気軽に素地指導として、もっと低学年の子どもに扱わせることができるでしょう。

技能

・ Counting　数を数える。

・ Graphing　平面を格子状の微小部分に分け、絵を表現する。

年齢

・ 7 歳以上

教材

・OHP の原本から作られた数字による色（16 ページ）の OHP シート。

＜子どもごとに必要なもの＞

・ワークシート学習 : 子どもファクシミリ（17 ページ）

・ワークシート学習 : 自分の絵を描こう（18 ページ）

色を数で表す

問いかけの例

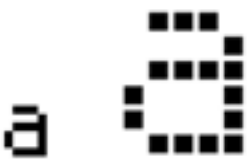
① ファクシミリ（FAX）は何をするための機械でしょう？

② コンピュータはどうやって画像を保存しているのでしょう？（絵を描くプログラム、グラフィックスを使ったゲーム、マルチメディアシステム）

③ 数しか扱えないコンピュータはどうやって絵を保存しているのでしょう？

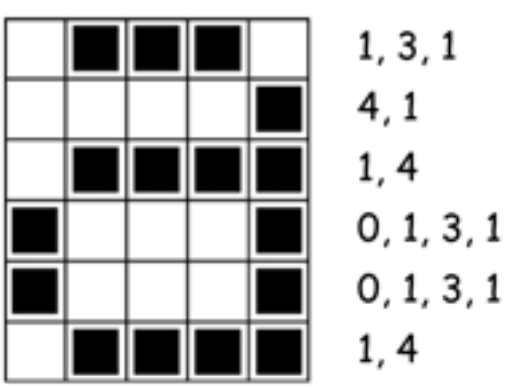
（この学習のために、子どもにファクシミリを使わせてみるのもよいでしょう）

説明シートを使った実演



コンピュータの画面はピクセルと呼ばれる小さな点で縦横に区切られています。白黒の絵では、それぞれのピクセルは白か黒の色になります。

上では「a」という字をピクセルが見えるように拡大しています。コンピュータが絵を記録するときは、どの点が黒でどれが白かを記録しています。



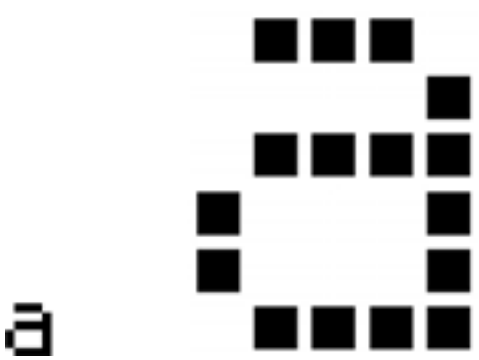
上の図は、絵を数字で表す様子です。1 行目を見ると、1 個の白に続いて 3 個の黒が並び、1 個の白があります。そこで、この行には 1,3,1 と書いてあります。

最初の数字は、白のピクセルの数を表しています。最初のピクセルが黒だったときは、その行は 0 で始まります。

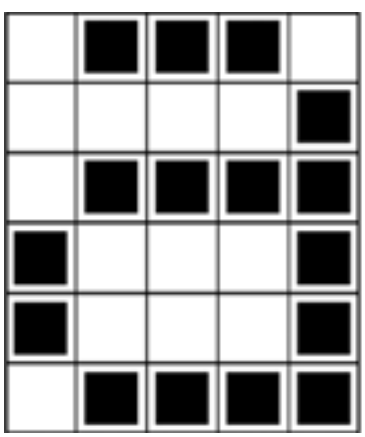
17 ページのワークシートでは、子どもは今説明したやり方を使って、コードから絵を描くことができます。

解説シート

色を数で表す

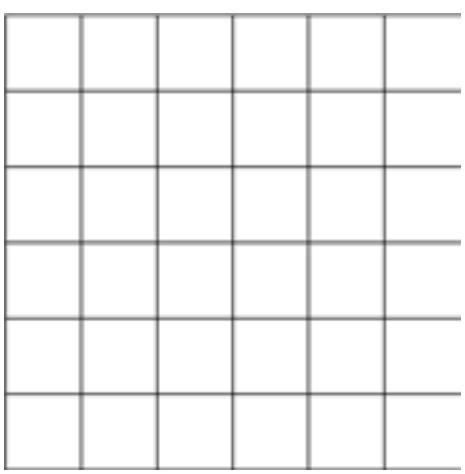


▲コンピュータの画面に表示された「a」と、
ピクセルが画像を作っている様子を示す拡大図



1, 3, 1
4, 1
1, 4
0, 1, 3, 1
0, 1, 3, 1
1, 4

▲同じ画像を数値でコード化した

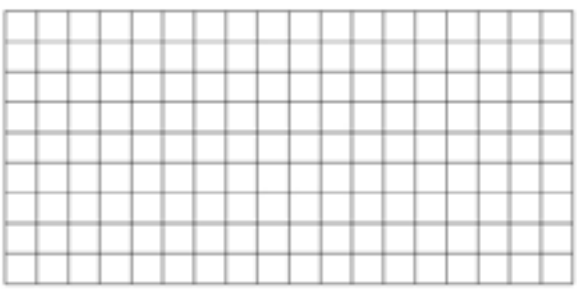


▲空のマス目（説明用）

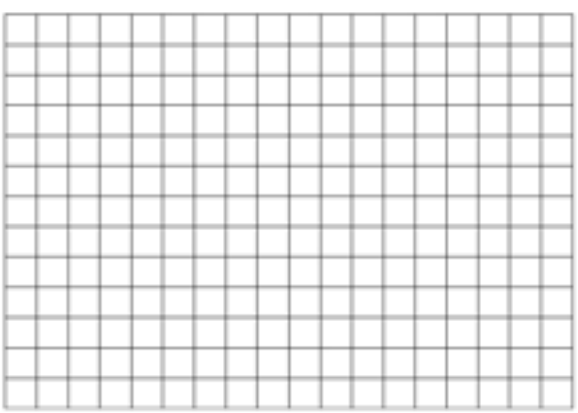
ワークシート

子どもファクシミリ

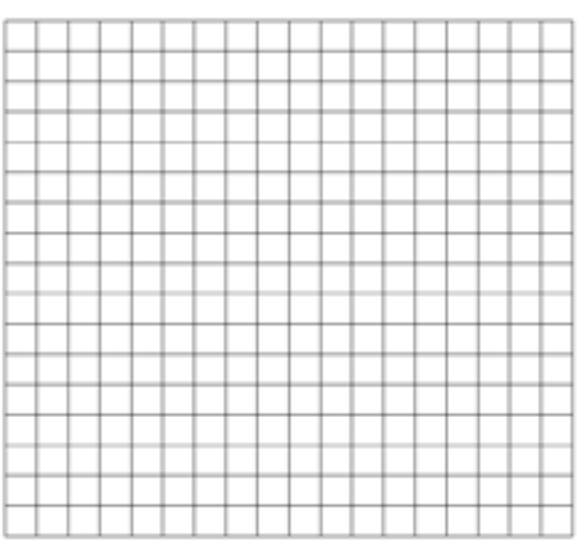
最初の絵はいちばん簡単で、最後の絵はいちばん複雑です。間違いやすいので、片手に色を塗るための鉛筆を、もう片方の手に消しゴムを持つとよいでしょう！



4, 11
4, 9, 2, 1
4, 9, 2, 1
4, 11
4, 9
4, 9
5, 7
0, 17
1, 15



6, 5, 2, 3
4, 2, 5, 2, 3, 1
3, 1, 9, 1, 2, 1
3, 1, 9, 1, 1, 1
2, 1, 11, 1
2, 1, 10, 2
2, 1, 9, 1, 1, 1
2, 1, 8, 1, 2, 1
2, 1, 7, 1, 3, 1
1, 1, 1, 1, 4, 2, 3, 1
0, 1, 2, 1, 2, 2, 5, 1
0, 1, 3, 2, 5, 2
1, 3, 2, 5

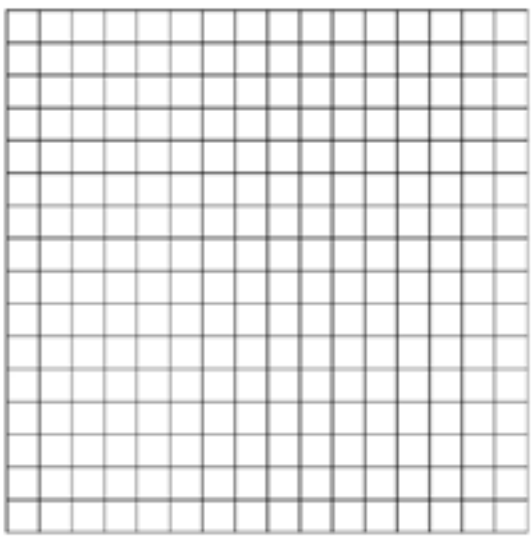


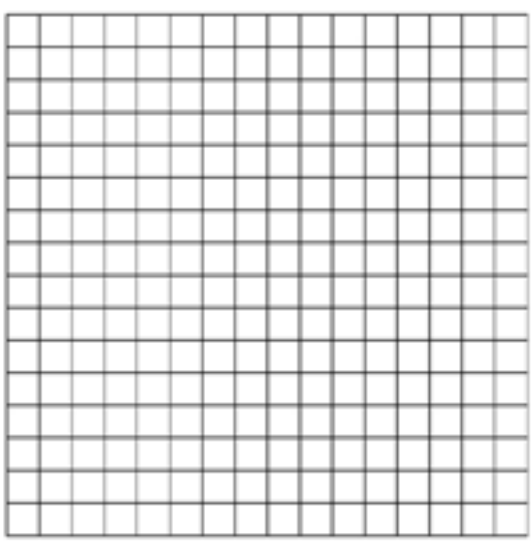
6, 6
5, 1, 2, 2, 2, 1
6, 6
4, 2, 6, 2
3, 1, 10, 1
2, 1, 12, 1
2, 1, 3, 1, 4, 1, 3, 1
1, 2, 12, 2
0, 1, 16, 1
0, 1, 6, 1, 2, 1, 6, 1
0, 1, 7, 2, 7, 1
1, 1, 14, 1
2, 1, 12, 1
2, 1, 5, 2, 5, 1
3, 1, 10, 1
4, 2, 6, 2
6, 6

ワークシート

自分の絵を描こう

絵が数で表せることがわかったところで、自分で描いた絵を友だちにあげましょう。まず、上のマス目に絵を描きます。次に、下のマス目の隣にコードを書きましょう。そして切り離したら、下のほうを友だちに渡して色を塗ってもらいましょう。（もし必要がなければ、マス目全体を使う必要はありません。マス目の下のほうは使わなくても構いません。「,」をきちんと入れましょう）





ワークシート

自分の絵を描こう（応用問題）

もし色の付いた絵を描きたければ、色を数で表すこともできます（たとえば、0 が白、1 が黒、2 が赤など）。そして、2 個の数字でピクセルの並びを表しましょう。1 個目の数字は並んでいる数を表し、2 個目は色を表します。色の付いた絵を描いて、友だちに渡しましょう。友だちには、どの数がどの色なのかを教えてね！

ワークシートの使い方の工夫

1. 上のマス目に絵を描くときにトレーシングペーパーを置いてから描くと、絵が完成したときにマス目なしで絵を見られます。
2. 大きなマス目を作れば、子どもは塗りつぶすだけでなく、付箋紙を貼ったり、物を置くことができるかもしれません。

発展課題

ピクセルの並びを2進数で表すときは、長さに上限があるのが普通です。7個までしか表現できない場合には、どうやって12個の並びを表現したらよいでしょう？（たとえば、7個の黒と0個の白、5個の黒のように、7個ずつに分ける方法があります）

実際のコンピュータでは

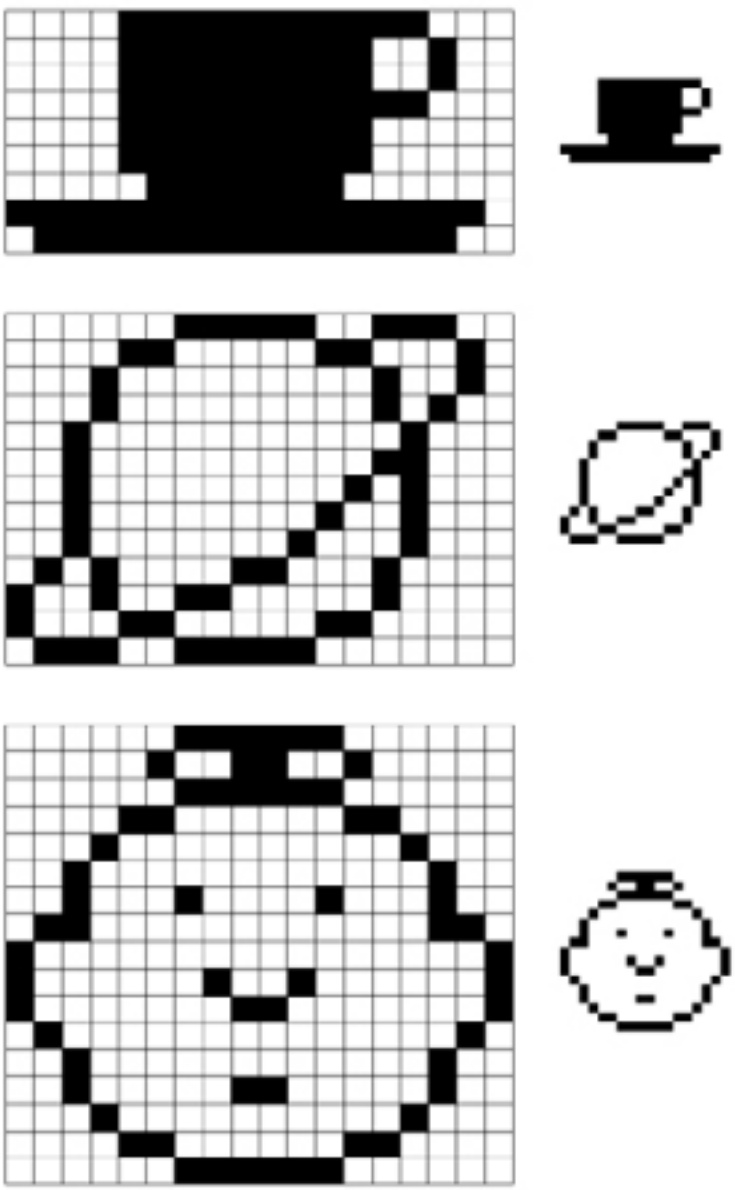
ファクシミリの機械は単純なコンピュータそのものです。白黒のページを 1000×2000 程度のピクセルに読み取り、他のファクシミリの機械にモデムで送り、先方はピクセルを紙に印刷します。ファクシミリの画像には、大きな白（たとえば余白）や大きな黒（たとえば罫線）の塊が存在します。色の付いた画像でも、たくさんの繰り返しがあります。このような巨大な画像を少しでも多く記録するために、コンピュータではさまざまな圧縮技法が使われています。もし画像の圧縮を行わないと、転送するために多くの時間がかかりますし、記憶容量も多く必要になります。そうすると、時間がかかりすぎてファクシミリで通信したり Web で画像を見ることが難しくなってしまうかもしれません。たとえば、ファクシミリの画像は、通常 1/7 程度に圧縮されています。圧縮がないと、7 倍の時間がかかってしまうのです！

写真や絵は 1/10 から 1/100 に圧縮されることがあります（それぞれ異なる技術を使っています）。これらの技術によって、ディスクには多くの画像を記憶できますし、Web の表示にかかる時間も節約できます。

大量のデータを圧縮するためにはいくつかの方法があります。次の章でそのひとつを紹介します。

解答とヒント

子どもファクシミリの答え



学習 3　それ、さっきも言った！

テキスト圧縮

概要

コンピュータが情報を格納できる量は限られているので、情報はできる限り効率的に表現される必要があります。これは圧縮と呼ばれます。情報を記録する前に圧縮し、取り出された後に戻すことで、コンピュータはより多くのデータを記憶したり、それらをインターネットに高速に送ることができます。

教科学習との関連

・English: Recognising patterns in words and text.

※［国語］言葉や文章のパターンを認識する

・Technology: Technological knowledge and understanding. How computers work.

※［技術］コンピュータの働きについての、工学的な知識と理解。

技能

・Copying written text　文章を読み取って写す。

年齢

・9 歳以上

教材

・OHP シート：説明用（25 ページ）

＜子どもごとに必要なもの＞

・ワークシート：それ、さっきも言った！（26 ページ）

・ワークシート：応用問題（27 ページ）

・ワークシート：簡潔に（28 ページ）

・ワークシート：応用問題（上級者用）（29 ページ）

それ、さっきも言った！

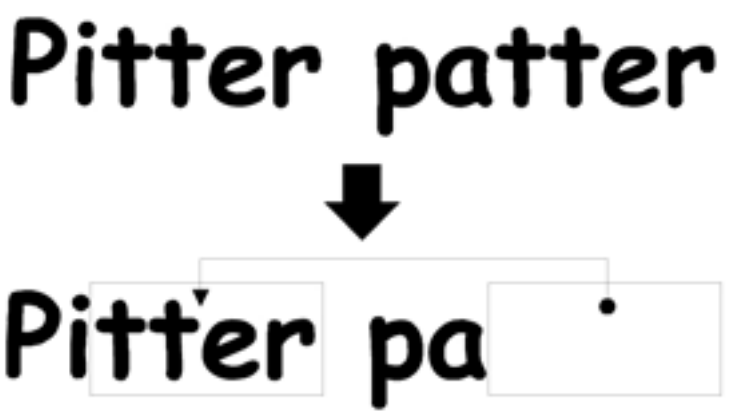
説明

コンピュータは大量のデータを記憶したり転送したりする必要があります。しかし、記録できる量は限られていますし、そのままではインターネットで送るのに時間がかかるので、テキストを圧縮することが行われています。

実演と議論

「The Rain」の説明（25 ページ）を見て、この詩に含まれる文字のパターンを探してみましょう。繰り返し出てくる 2 文字以上の文字の並び、または単語か文を見つけられますか？（下の図のように箱で置き換えてみましょう）

カードを作って重ねてみるとわかりやすいです。



解説シート

それ、さっきも言った！

1: The Rain

2: Pitter patter

3: Pitter patter

4: Listen to the rain

5: Pitter patter

6: Pitter patter

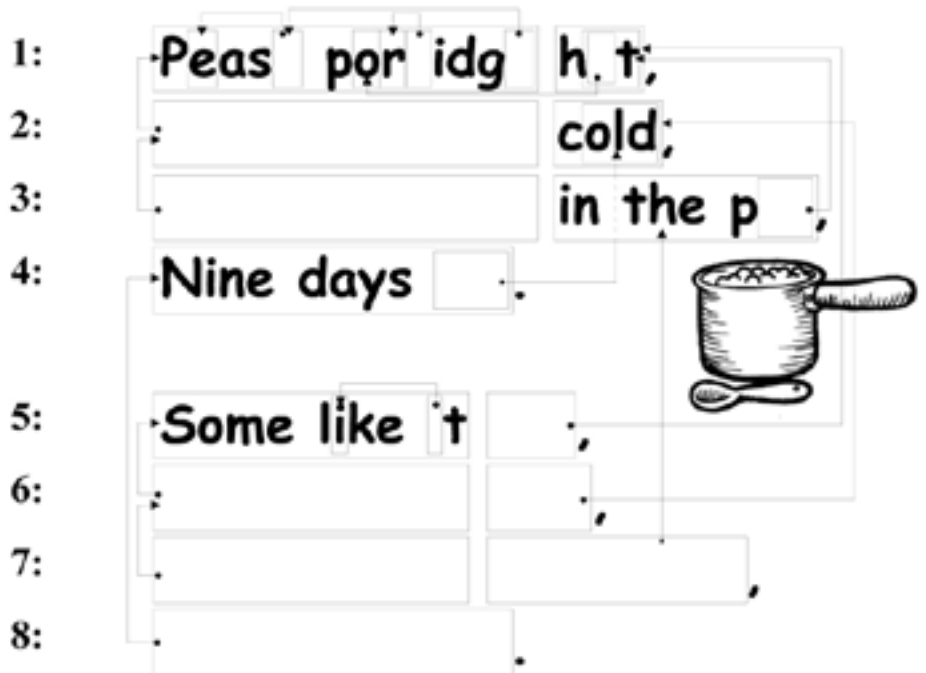
7: On the window pane

rain:雨、pitter patter:雨の音、
window pane:窓ガラス

ワークシート

それ、さっきも言った！

詩の中から、単語や文字がたくさん抜けています。抜けている単語や文字を戻してみましょう。 矢印が指している空白（箱）から見つけることができるでしょう。



次に簡単な詩か童謡を選んで自分のパズルを作ってみましょう。矢印は文章の前のほうを指すように注意します。元の文章は、普通に読むように左から右に、上から下に書かれている必要があります。

チャレンジ：元の単語をどこまで減らせるか試してみよう！

「Three Blind Mice」「Mary Mary Quite Contrary」「Hickory Dickory Dock」などのマザーグースの詩や、スース先生の本も試してみてください。

ヒント：矢印でごちゃごちゃしないように、文字や単語を書くときは周りに余白を作りましょう。そうすれば箱の中に箱を描いたり、それに矢印を付けることができます。

最初に詩を書いてから箱の位置を考えると、パズルを考えるのが楽になります。

ワークシート

応用問題

このパズルは解けますか？

空欄が空欄の一部を指すこともあります。このようなときも、左から右に文字を写せば元の文章に戻ります。コンピュータでも、この方法は文字やパターンが長く続く場合の圧縮に使われています。

自分でも作ってみましょう。

コンピュータでは、箱と矢印は数字で表現されます。たとえば、

Banana

は Ban□2,3）と表現できます。「2」は文字を写すために、Ban の後ろから 2 文字戻る意味です。

この場合は「a」の文字の場所です。

Ban___

そして「3」はその場所から順に 3 文字写すことを意味します。最初に「a」を 2 文字右に写して、次に「n」を 2 文字右に写して、最後に「a」を 2 文字右に写します。

Bana__

Banan_

Banana

この方法では単語をコード化するために 2 つの数字を使うので、2 文字以上の繰り返しには圧縮効果がありますが、それ以外にはありません。実際、1 文字の繰り返しに使った場合にはファイルのサイズが大きくなってしまいます。

コンピュータが圧縮できるような単語を考えてみよう。
友だちは元に戻せるかな？

ワークシート

簡潔に

単語はいくつ必要？

コンピュータになったつもりで、できるだけ文字を減らしてみよう。すでに出てきた2文字以上の組は線で消そう。もう矢印で指されることはないのだから。できるだけ多くの文字を消すのが目標です。

1: I know an old lady who swallowed a bird
2: How absurd! She swallowed a bird!
3: She swallowed the bird to catch the spider
4: That wriggled and jiggled
5: and tickled inside her
6: She swallowed the spider to catch the fly
7: I don't know why she swallowed a fly
8: Perhaps she'll die...

ワークシート

応用問題（上級者用）

難しい圧縮に挑戦してみる？

下のお話をコンピュータで処理すると、少なくとも 1,633 個の文字が消されることがわかりました。あなたは何個見つけられますか？ 繰り返し出てくる２文字以上の文字を消せるんですよ。がんばって！

Once upon a time, long, long ago, three little pigs set out to make their fortunes. The first little pig wasn't very clever, and decided to build his house out of straw, because it was cheap. The second little pig wasn't very clever either, and decided to build his house out of sticks, for the "natural" look that was so very much in fashion, even in those days. The third little pig was much smarter than his two brothers, and bought a load of bricks in a nearby town, with which to construct a sturdy but comfortable country home.

Not long after his housewarming party, the first little pig was curled up in a chair reading a book, when there came a knock at the door. It was the big bad wolf, naturally.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the first little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he did huff, and he did puff, and the house soon collapsed. The first little pig ran as fast as he could to the house of sticks, and was soon safe inside. But it wasn't long before the wolf came calling again.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the second little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he did huff, and he did puff, and the house was soon so much firewood. The two terrified little pigs ran all the way to their brother's brick house, but the wolf was hot on their heels, and soon he was on the doorstep.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the third little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he huffed, and he puffed, and he huffed some more, but of course, the house was built of brick, and the wolf was soon out of breath. Then he had an idea. The chimney! He clambered up a handy oak tree onto the roof, only to find that there was no chimney, because the third little pig, being conscious of the environment, had installed electric heating. In his frustration, the wolf slipped and fell off the roof, breaking his left leg, and severely injuring his pride. As he limped away, the pigs laughed, and remarked how much more sensible it was to live in the city, where the only wolves were in the zoo. And so that is what they did, and of course they all lived happily ever after.

実際のコンピュータでは

コンピュータの記憶容量は信じられない勢いで増えています。過去 25 年間で、ディスク容量は 100 万倍になりました。しかし、コンピュータに入れたい情報も増え続けています。コンピュータは本を丸ごと、それも図書館の本すべてや、音楽、映画を含めて記録することができます。巨大なファイルは今でも、インターネットでダウンロードに時間がかかる問題があります。今後、コンピュータは携帯電話や腕時計の大きさに、たくさんの情報を蓄えるようになることでしょう！

この問題にはすでに解決策があります。しかし、より大容量の記憶装置を買ったり速いモデムを買う代りに、私たちはデータを圧縮することで必要な領域を少なくすることもできます。このような圧縮と伸張の処理は、コンピュータが自動的に行うことが一般的です。私たちはディスクにたくさんのデータを入れられることや Web ページを速く表示できることを知っていますが、実際にはコンピュータはたくさんの処理をしているのです。

数多くの圧縮技術が発明されています。今回の学習で使った、すでに出現したテキストのかたまりをポインタで指す圧縮技術は、2 人のイスラエルの教授たちにより 1970 年代に発明されました。「Ziv-Lempel 符号」または「LZ 符号」と呼ばれています。この技術はどんな国の言語にも有効で、サイズも半分以下にできます。パーソナルコンピュータでは ”zip” と呼ばれ、” GIF” 画像や高速モデムでも使われています。モデムでは、電話回線で転送するデータ量を少なくすることで、高速化を実現しています。

他の圧縮技術では、たくさん使われる文字のコードを他の文字より短くするアイデアが使われることがあります。モールス符号などで使われています。

解答とヒント

それ、さっきも言った！（26 ページ）

Pease porridge hot,
Pease porridge cold,
Pease porridge in the pot,
Nine days old.

Some like it hot,
Some like it cold,
Some like it in the pot,
Nine days old.

学習４　カード交換の手品

エラー検出とエラー訂正

概要

データをディスクに格納したり他のコンピュータとやり取りするときに、私たちはデータが変化しないことを期待しています。しかし、たまにデータが偶然変化してしまうことがあります。

この章の学習では、データの破損（エラー）を検出して訂正するために、手品を使います。

教科学習との関連

・Mathematics: Number Level 3 and up. Exploring computation and estimation.

※［数と計算］（小４）計算と検算（確かめ）

・Algebra Level 3 and up. Exploring patterns and relationships.

※［数量関係］パターンや関係を探求する

技能

・Counting　数を数える。

・Recognition of odd and even numbers　偶数と奇数の認識。

年齢

・9 歳以上

教材

・片面のみに色がついた（冷蔵庫に貼るような）マグネットシート 36 枚

・デモンストレーションに使うためのホワイトボードやマグネットボード

＜子どものペアごとに必要なもの＞

・片面のみに色がついた同一のカード 36 枚

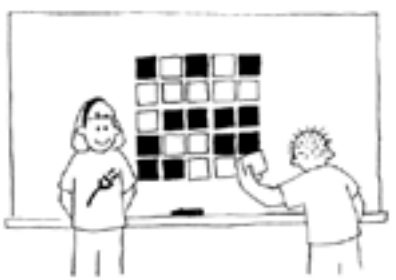
手品

実演

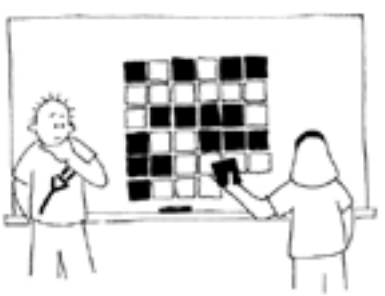
手品師になるチャンスですよ！

どちらの面かわかるカードがたくさん必要です。大きなシートを切って使うときは、片面だけ色がついたものを使いましょう。両面の色が違う平らな磁石のカードを使うと便利です。

① 子どもを選び、カードを好きな面が見えるように5行5列の正方形に並べさせます。



「もっと難しくしてみましょうか」と言いながら、
何気なく縦と横に1列ずつ加えます。



この加えたカードが手品のポイントです。
あなたは縦と横の色のついたカードがそれぞれ偶数になるように
カードを加える必要があります。

② 目をつぶっている間に、子どもにカードを1枚だけ裏返してもらいましょう。
そのカードの行と列は色のついた枚数が奇数になるので、裏返されたカードを見分けることができます。

子どもはこの手品のタネがわかりましたか？

手品の説明

① ２人組になって、５行５列にカードを並べましょう。

② 色のついたカードは、縦と横にそれぞれ何枚ずつありますか？
奇数でしょうか偶数でしょうか？（0 は偶数です）

③ 次に、横の色のついたカードが偶数枚になるように、右に６枚目のカードを置きましょう。
このカードのことをパリティカードといいます。

④ いちばん下に６枚目のカードを置いて、
縦の色のついたカードが偶数枚になるようにしましょう。

⑤ 次に、カードを１枚裏返します。縦と横はどうなりましたか？
（色のついたカードの枚数が奇数になりました）
パリティカードは間違いが起きていることを知るために使われます。

⑥ かわりばんこに手品をしてみましょう。

発展学習

① 他のものでもやってみましょう。２つの状態を持つものなら何でもいいです。
例えば、トランプやコイン（裏か表か）または、１か０が書かれているカードなどは２進法に関係させることができます。

② もし２枚以上のカードがひっくり返されたらどうなるでしょう。
（何かが変えられたことはわかるとしても、どの２枚がひっくり返されたかを正確に知ることはできません。たいていは、「2 組のカードのどちらか」に絞り込むことができます。4 枚がひっくり返されたときは、すべてのパリティが打ち消しあって、エラーがまったく検出できないこともあります）

③ もうひとつ面白い課題を、パリティカードが交わる右下の角のカードで考えてみましょう。もしあなたが、その列方向で正しいパリティとなるようにカードを置いたら、その左の行方向のカードに対しても正しいものとなっているでしょうか？（その答えは、いつもイエスです）

④ このカードを使った学習では、色がついた偶数枚のカードを偶数パリティ（1 のビットを偶数個使用するコード）として扱いました。奇数パリティでもできるでしょうか。
（可能です。ただし、右下のカードについては同数の行と列で作られている場合に限ります。行と列の数が両方とも偶数か両方とも奇数である必要があります。たとえば、5×9 や 4×6 の並びのときはうまくできますが、3×4 の場合はできません）

エラーチェックの実例　ISBN コード

値をチェックする方法は、本の番号にも使われています。書店に並んでいる本は、たいていカバーに 10 桁か 13 桁の数字の番号が書いてあります。一番最後の数字は、パリティのところでやったように、チェックのための数字です。

本の注文などでこの ISBN（国際標準図書番号）を使うときは、出版社はチェックの数字を確認して、本を間違えないようにしています。（せっかく注文したのに、間違った本が届いたら残念ですからね）

では、10 桁のチェックサムを計算してみましょう。最初の数字に 10 を掛けます。2 番目に 9、3 番目に 8、というように 9 番目の数字に 2 を掛けるまで続けます。それぞれの計算した数字を全部足し合わせます。

例えば、ISBN 0-13-911991-4 の場合は次のようになります。

□0 × 10□ + □1 × 9□ + □3 × 8□ + □9 × 7□ + □1 × 6□
+ □1 × 5□ + □9 × 4□ + □9 × 3□ + □1 × 2□
= 172

でた数字を 11 で割りましょう。

172 ÷ 11 = 15 あまり 7

余りが 0 なら答えは 0 とします。そうでなければ、11 からその余りを引きます。

11 − 7 = 4

見てください。この数字は、ISBN の最後の数字と同じですね！ もし ISBN の最後の数字が 4 でなかったら、何か間違いがあったことがわかります。

ところで、余りが 10 のときは 1 桁で表すことができません。こんな場合は、文字 X（ローマ数字の 10）を使います。

数字をチェックするもうひとつの例として、食料品に使われているバーコードがあります。これは、違った方式で計算しています。もし、バーコードが間違って読まれたら、最後の数字は計算した結果と違ってしまいます。そんなときは、読み取り機がピーと鳴って知らせ、レジの人がもう一度そのバーコードを読み取り直します。

チェックサムは正しかった？

時々、エラー（間違い）があります。
よくあるエラーは、このようなものです。

- 数字が変わっていた
- 隣どうしを入れ替えてしまった
- 余計な数字が入っていた
- 数字が抜けていた

チェックサムの 10 を示す文字 X（ローマ数字）を使った本を見つけられましたか？
きっとすぐに見つけられますよ。11 冊に 1 冊はそうなのですから。

見抜けないエラーにはどんなものがありますか。
チェックサムの値を変えずに数字を変えられるでしょうか？
2 つの数字が入れ替わったとしたらどうでしょう？

実際のコンピュータでは

千円を自分の銀行口座に預けることを想像してみましょう。銀行の窓口では預金額を入力します。そして、その額は中央のコンピュータに送られます。しかし、その金額が送られている間に通信が故障して、千円を示すコード番号が10万円に変わったとしたらどうでしょう。あなたが預金者なら何の問題もありませんが、銀行側にすれば明らかに問題です。

転送するデータのエラーを見つけることはとても重要です。だから、受け取る側のコンピュータは送られてくるデータが送信線（電線）上の電気的な障害で壊れてないかを調べる必要があります。エラーが送られた時は元のデータを再び送りますが、それが不可能な場合もあります。たとえば、元のデータが入っていたテープやディスクが故障してしまった場合です。

もしデータを遠い宇宙探査機から受け取る場合には、エラーが起きたからといって再送信されるのを待つことはとても時間がかかります。(たとえ、地球から一番近づいたときでも、木星から送られてくる電波は30分以上かかります)

私たちは、データが壊れているかどうかを確認して（エラーの発見)、元のデータを再現できる必要があります（エラーの訂正)。

カードを裏返す手品と同じ仕組みは、コンピュータの中でも使われています。仮想的な行と列にビットを置いて、それぞれの行と列にパリティビットを加えることで、エラーの発生と起きた場所がわかります。そして、誤ったビットを戻すことでエラーを訂正することができるのです。

実際には、複数のエラーを検出して修復するための複雑なエラー制御システムが使われています。たとえばコンピュータのハードディスクでは、ディスクの一部が壊れても動き続けられるように、エラーを直すための領域をたくさん割り当てています。この仕組みはパリティチェックの方式と関係しています

解答とヒント

1個の数字の値が増えて、1個の数字の値が減る場合には、合計が変わらないので見抜けません。

学習５　２０の扉

情報理論

概要

1000 ページの本にはどれくらいの情報があるのでしょう。1000 ページの電話帳にはもっと情報があるのでしょうか、また、同じ 1000 ページでも白紙を綴じただけのもの、あるいは、トルーキンの『指輪物語』はどうでしょう。もし、そんなことが測れるなら、情報を蓄えるにはどれくらいの場所が必要か見積もることができます。たとえば、飛行機に荷物を預けると、荷物に NRT、HND、HKG などと書いた紙が付けられます。たった 3 文字のローマ字ですが、だいたいどの空港かを想像することができますね。ちなみに答えは、成田、羽田、香港です。では、次の母音字を欠いた英語の文を読むことができますか？

Ths sntnc hs th vwls mssng.

きっと「This sentence has the vowels missing.」と正しく読めると思います。つまり、母音にはそれほど情報はないのです。ここでは、情報量を測る方法について学びます。

教科学習との関連

・Mathematics: Number Level 3 and up.

※［数と計算］数の比較：より大きい、以上、以下、未満、範囲。数の大小は小 1 から扱っていますが、「≧」は、数と式の領域で中 1 に配当されています。

・Algebra Level 3 and up.

※［数量関係］パターンによって数を表すこと、また高校での等比数列にも関わりを持たせることができます。

・ English

※［国語］言葉や文章の持つ一種の冗長性を意識するのに有用な教材です。

技能

・Comparing numbers and working with ranges of numbers　範囲が定まった数を扱う。

・Deduction　推論する。

・Asking questions　質問をする。

年齢

・10 歳以上

教材

・はじめの活動には何も必要ありません

＜発展学習として子どもごとに必要なもの＞

・ワークシート学習：決定木（ 40 ページ）

20の扉

話し合おう

情報とは何かを話し合ってみましょう

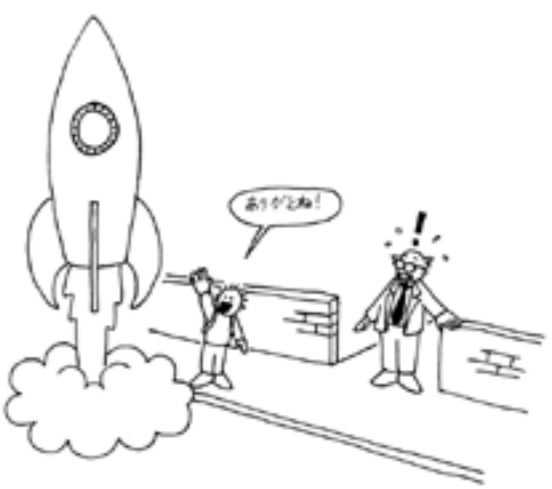
本に書いてある情報量はどうやって測れるでしょう。ページ数が重要ですか、それとも字数でしょうか。 ある本に他の本よりも多くの情報を含ませることができますか。すごく退屈な本や、逆に面白い本もあります。「デタラメ、デタラメ、デタラメ」という文だけが含まれている400ページの本があったときに、この情報量は電話帳より多いのでしょうか。

コンピュータ科学者は、文章や本にどれだけ驚きがあるかで情報量を測っています。既に知っていること、たとえばいつも歩いて学校に来る子が「きょうは歩いて学校にきたよ」と言ったとすると、これには情報量がありません。びっくりすることではないからです。でも、「きょうはヘリコプターで学校にきたよ」と言ったとしたら、これにはびっくりするでしょう。つまり、多くの情報量をもたらしたということになります。

では、メッセージの「びっくり度」はどうやって測るのでしょう。

ひとつの方法は、情報を推測するのがどれだけ難しいかをみることです。もし、いっしょに歩いて学校にきた友だちに「きょうどうやって学校にきたかあててごらん」と言われたら、1回で当てることができます。でも、以前ヘリコプターに乗ってきたことがあれば、「まさかヘリじゃないよね」と確かめる必要があるかもしれません。ましてや、宇宙船で旅をしたことがあるなら、もっと手間がかかるかもしれません。

メッセージの持つ情報量は、それを推測するのがやさしいのか難しいのかで測ります。次のゲームは、このようなことを理解するのに役立ちます。



ゲーム　20 の扉

ここでは 20 の扉というゲームを紹介します。子どもたちは、１人の子どもに質問をしていきます。その子は「はい」、「いいえ」しか答えられません。「はい」か「いいえ」で答えられるものであれば、どんな質問をしても構いません。

質問の例

今から当てて欲しい数が何種類かあります。

- 1 から 100 までの数です
- 1 から 1000 までの数です
- 1 から 1,000,000 までの数です
- 整数です
- 5 個の数がある順に並んでいます。
  数を順番に当ててください（例 2、4、6、8、10）

当てるのに何回質問する必要がありましたか。この回数を「情報量」といいます。

補足の話し合い

どのような作戦で質問していきましたか。どれがいちばんよかったですか。

1 から 100 までの数は 7 回の質問で当てることができます。それぞれの質問で範囲が半分になるようにすればよいのです。例を見てみましょう。

50 以下？	はい
5 以下？	いいえ
37 以下？	いいえ
43 以下？	はい
40 以下？	いいえ
41 以下？	いいえ
42 ですか？	はい！

面白いことに、範囲が 1000 までに増えたとしても、この回数は 10 倍になりません。質問するべき回数は、たった 3 回増えるだけです。範囲が 2 倍になるたびに、答えを見つけるための質問の回数は 1 回ずつ増えるだけなのです。

発展：メッセージにどれだけの情報があるだろう？

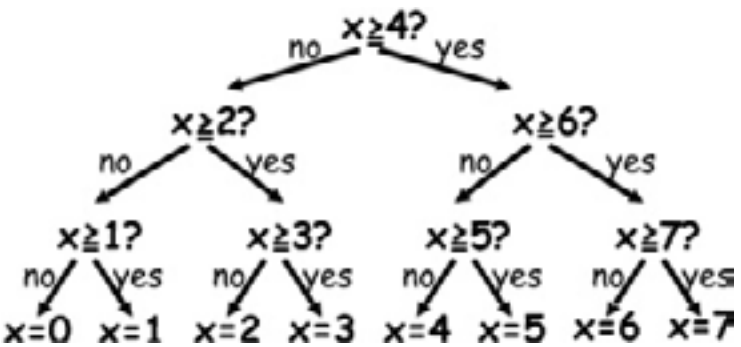
文字を当てるゲームをしましょう。先攻の人は 17 字までの意味のある文を作り、4 個以内の伏せ字■を入れます。濁点、半濁点は無視して「はんに■は■■えた」のように「あ」から「ん」までのひらがなだけを使います。後攻の人は、「初めの伏せ字は『ん』ですか」「2 番目の伏せ字は、『あ行』（あいうえお）ですか」「3 番目の伏せ字は、50 音順で、『た』よりも前ですか」など、「はい」か「いいえ」で答えられるような質問をしていきます。先攻の人は、どの伏せ字にどんな質問があって何と答えたかを記録します。伏せ字を全部推測し終えたら先攻後攻が変わります。さて、どこの質問の方法がいちばんわかりやすかったでしょう。（例題の答えは、「犯人はお前だ」）

ワークシート

決定木

実際に質問をしてみなくても、質問をしていく作戦を表すことができます。

下のような図は、「決定木」と呼ばれています。この図は 0 から 7 までの整数を当てるための決定木です。「はい」が yes で、「いいえ」が no です。



5 を当てるためには、どのような yes/no の決定が必要ですか。

ある数を当てるためには、何回の決定をする必要があるでしょう。

すてきなことを教えましょう。決定木のいちばん底の部分にある 0、1、2、3… の数の下に、2 進法（学習 1 参照）でその数を書いてみましょう。

木をよく見て下さい。「いいえ」を 0、「はい」を 1 で表したとすると、この木から何かが見つかるはずです。

数を当てるゲームでは、答えの並びがちょうど数を表しているように質問を選んで、並べているのです。

0 から 15 までの整数を当てるための、決定木を自分で作ってみましょう。

応用問題

誰かの年齢を当てるには、どのような決定木を使いますか。

また、「伏せ字」を当てるには、どうでしょう。

実際のコンピュータでは

有名なアメリカの数学者であるクロード・シャノンは、ジャグラーや一輪車乗りとしても有名です。そして、ここで学んだゲームのような実験を数多く手掛けました。彼は情報量をビット（「はい・いいえ」による決定の回数）で測りました。メッセージの情報量は、既に知っていることが何であるかによって違ってくることを発見しました。ある質問をすることによって、他の質問をしないでよくなることは、よくあることです。この場合、しないでも良くなったメッセージに含まれる情報量は少ないのです。たとえば、コイン投げの結果は、裏か表かですから、普通１ビットです。でも、コインが「いかさま」で、10回中９回も表が出るなら、その情報は１ビットよりも少なくなります。どうやって、１回のコイン投げの結果を１回未満の質問で知ることができるのでしょう。単純です。次のような質問をすればよいのです。「2回のコイン投げの結果はどちらも表だったか？」。いかさまコインの結果を書き並べると、こういった「表表」になることは、だいたい80％の回数で起こります。20％は「いいえ」なので、この場合はもっと質問をする必要があります。でも平均してみれば、コイン投げの回数よりも少ない回数で、いかさまコイン投げの結果を当てることができるでしょう。

シャノンは、メッセージの情報量のことを「エントロピー」と呼びました。「エントロピー」は、事象の数、つまりコイン投げの場合は、表・裏なので２だけではなく、それが起こる「確率」にも影響を受けます。ありえない出来事、つまりびっくりするような情報は、そのメッセージを当てるのに多くの回数の質問が必要です。ちょうど学校に来るのにヘリコプターを使ったときのように、それまで知らなかった情報をつけ加えているのですから。

メッセージのエントロピーは、コンピュータ科学者にとって大変重要です。エントロピーよりも小さな大きさにメッセージを圧縮することはできません。また、最良の圧縮法は、数を当てるゲームと同じです。答えを何ビットかの情報として記憶しておくことで、決定木を使ってコンピュータのプログラムは推測を行なうことができます。だから、情報を元のように再構築することができるのです。圧縮システムの中には、テキストファイルのファイルサイズを４分の１にまで小さくできるものもあります。これは、記憶容量の節約に大いに役立っています。

数を当てるゲームの方法は、コンピュータインタフェースにも使われています。利用者が次にどのキーを押すかを予想して候補を表示することにより、身体に障害があってキー操作が不自由な人でも平均で２回の「はい・いいえ」の答えだけでローマ字１字を打つことができます。この仕組みは携帯電話の文字入力にも使われています

解答とヒント

1回の「はい・いいえ」で答える回答は1ビットの情報に対応しています。その質問が「50よりも大きいですか」といった単純なものか、「20と60の間ですか」といったもう少し複雑なものであるかには関係しません。

数を当てるゲームでは、特定の方法で質問を選んでおくと、答えの並びを2進法で表したものと一致します。3は2進法では011ですが、決定木での質問を並べると「いいえ、はい、はい」になります。「いいえ」の代りに0、「はい」の代りに1を書いたときの結果は、3を2進法で表したものと同じです。

年齢を当てる決定木は、小さな数の方へ片寄るでしょう。

文の中の伏せ字を当てる決定木は、前の文字が何であるかに関係します。これは英語の場合、どの文字の次にどの文字が来やすいかという統計が研究として確立しているからです。日本語の場合にも、前後にある既知の字と辞書を使って候補を絞ることができます。

第２部
コンピュータを働かせる
アルゴリズム

コンピュータを働かせる

コンピュータはあらかじめ用意された処理の手順に従って動作します。これらの手順により、コンピュータは情報を並べ替えたり、検索したり、送信することが可能になります。これらの処理を高速に行うためには、データを集合から検索したり送信したりするための効率のよい方法が必要になります。

ある仕事を行うための処理の手順をアルゴリズムと言います。アルゴリズムの考え方はコンピュータ科学で重要です。アルゴリズムを使って、私たちはコンピュータで問題を解くことができます。アルゴリズムには他のものより速いものがあります。アルゴリズムの多くは、それまで時間がかかりすぎて解けなかった問題を解決可能にしました。たとえば、πを100万桁計算したり、Webから自分の名前を検索したり、荷物をコンテナに入れる最適な方法を見つけたり、大きな数（たとえば100桁）が素数かどうかを判定することなどです。

学習６　戦艦

探索アルゴリズム

概要

コンピュータは大量のデータから情報を探すことを求められることが多く、そのために高速で効率の良い方法が必要です。今回の学習では、線形探索、二分探索、ハッシュ法という３種類の探索技術を扱います。

教科学習との関連

・Mathematics: Number Level 3 and up. Exploring numbers: Greater than, less than and equal to

※［数と計算］数の比較：より大きい、以上、以下、未満、範囲。数の大小に関しては小１から扱っていますが、学習指導要領で「≧」は、中１の数と式の領域に配当されています。

・Geometry Level 3 and up. Exploring shape and space: Co-ordinates

※［数量関係（中）］座標は中１で学びます。でも、このような扱いなら気軽に素地指導としてもっと低学年の子どもに扱わせることができるでしょう

技能

・Logical reasoning　論理的推論を行う。

年齢

・９歳以上

教材

＜子どもごとに必要なもの＞

・戦艦ゲームのコピー（生徒数の半分でよい）

1A、1B（ゲーム１用）
2A、2B（ゲーム２用）
3A、3B（ゲーム３用）

・予備のゲームシートである 1A’、1B’、2A’、2B’、3A’、3B’ も数枚のコピーが必要かもしれません。

戦艦　導入ゲーム

学習の概要

① 15 人くらいの子どもを教室の前に並ばせましょう。彼らには２桁の数字が書かれたカードを持たせます（値は 0 から 99 で、順序はランダムです）。他の子どもには数字は見せないようにします。

② 他の子どもには 4 個から 5 個のおはじきの入った箱を持たせます。彼らは与えられた数を探すのが仕事です。彼らは、おはじきを渡してカードを見せてもらうことができます。正しい数を見つけたときに手元に残っていたおはじきは自分のものになります。

③ 何回か繰り返してみましょう。

④ 次に、カードを切り直してもう一度配ります。今度は、子どもを自分たちで小さい順になるように並ばせます。そして数を探させます。

　数が順番に並んでいる場合には、真ん中の子どもにおはじきを渡して数を聞くだけで、半分の子どもを候補から除くことができます。この処理を続けると、必ず３個のおはじきを使うだけで数を見つけることができます。明らかに効率的です。

学習

子どもは戦艦ゲームを通して、コンピュータがどのように探索を行っているかを体験できます。ゲームの中で、船をどうやって見つければよいかを考えさせていきましょう。

戦艦　線形探索ゲーム

子どもに次の指示を伝えてください

① 自分たちで２人組を作ります。1 人は 1A のシートを、もう 1 人は 1B のシートを持ちます。自分のシートは相手に見せないでください！

② 両方とも、それぞれのゲームシートの上にある戦艦を 1 つ丸で囲み、相手にその番号を伝えます。

③ 次に、かわりばんこに相手の船がどこにいるかを推測します。（あなたは船の名前（英字）をいい、相手はその名前の船の番号を答えます）

④ 相手の船の場所を当てるまでに、何回攻撃しましたか？ それがあなたの得点です。少ないほうが勝ちです。

（シート 1A と 1B のほかに、もっとゲームをしたい子どもや間違って相手にシートを見せてしまった子どものために 1A' と 1B' が用意されています。また、2A、2B や 3A、3B の予備シートも用意されています。）

振り返りの議論

① 何点でしたか？

② いちばん小さい得点といちばん大きい得点は何点でしょう？
（子どもが同じ船を２回以上攻撃しないと仮定すると、それぞれ 1 と 26。この探索手法は 1 個ずつ調べていくため「線形探索」と呼ばれます）

※日本の教室で行われた線形探索ゲーム
相手のシートが見えないように、机の間を離し、筆箱で自分のシートを隠しています。

戦艦　二分探索ゲーム

指示

このゲームのやり方は前のゲームと同じですが、船の数字は小さい順に並んでいます。「さっきと並びが違うよ」など、このことを子どもに気づかせてから始めてください。

① 自分たちで２人組を作ります。１人は 2A のシートを、もう１人は 2B のシートを持ちます。自分のシートを相手に見せてはいけません！

② 両方とも、それぞれのゲームシートの上にある戦艦を１つ丸で囲み、相手にその番号を伝えます。

③ 次に、かわりばんこに相手の船がどこにいるかを推測します。（あなたは船の名前（英字）をいい、相手はその名前の船の番号を答えます）

④ 相手の船の場所を当てるまでに、何回攻撃しましたか？ それがあなたの得点です。少ないほうが勝ちです。

振り返りの議論

① 何点でしたか？

② 勝つためにはどうすればよいですか？

③ どの船を最初に選ぶべきだったでしょう？（真ん中の船はあなたに、左右どちらの半分に目的の船がいるかを伝えてくれます）次に選んだのはどれでしょう？（もう一度、あなたが選んだ半分の真ん中の船を選ぶのが最も良い戦略です）

④ この戦略を使うと、目的の船を見つけるのに何回の攻撃が必要ですか？（5 回以下）
この探索手法は問題を２つの部分に分けていくことから「二分探索」と呼ばれます。

戦艦　ハッシュ法を使ったゲーム

指示

① 前のゲームと同じようにシートを持ち、選んだ船の番号を相手に伝えなさい。

② このゲームでは、0 から 9 のどの列にその船がいるかを見つけます。船の番号の各桁を足し合わせた最後の桁が、船がいる列の番号になります。たとえば、2345 の船は 2+3+4+5 を計算して 14 になります。和の最後の桁は 4 なので、その船は 4 の列にあるはずです。列がわかれば、あとはその中のどの船かを予想するだけです。この手法は数字を混ぜ合わせることからハッシュ法と呼ばれています。

③ 次に、新しい探索戦略を使ってゲームをしましょう。別の列を選ぶことで、同じシートを使って複数回遊ぶことができます。
（このゲームでは、3A と 3B は必ず組にして使ってください。列に入る船が対応している必要があるからです）

振り返りの議論

① 得点を集めて議論しなさい。

② どの船が見つけやすかったですか？（列に 1 個だけある船）どの船が見つけにくかったですか？（たくさん入っている列にある船）

③ どの探索手法が速かったですか？それはなぜでしょう？

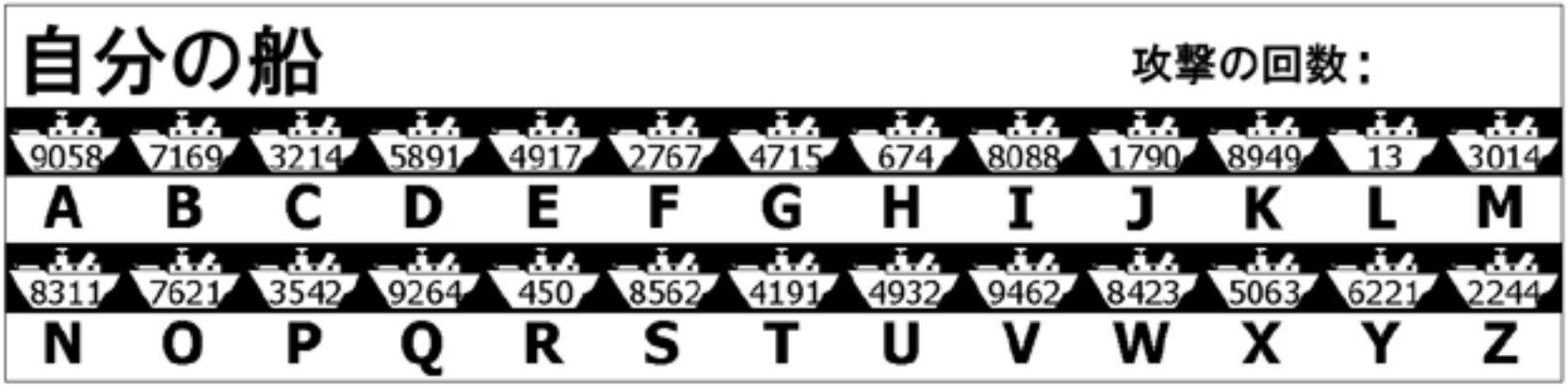
3 種類の探索手法は、それぞれどんなところが利点でしょう？
2 番目は 1 番目より速いですが、1 番目はソートしておく手間が不要です。
3 番目は他の 2 個より速いことが多いですが、ごくたまにとても遅いことがあります。
すべての船が 1 個の列に入ったときは最悪のケースで、1 番目と同じ速さになってしまいます。

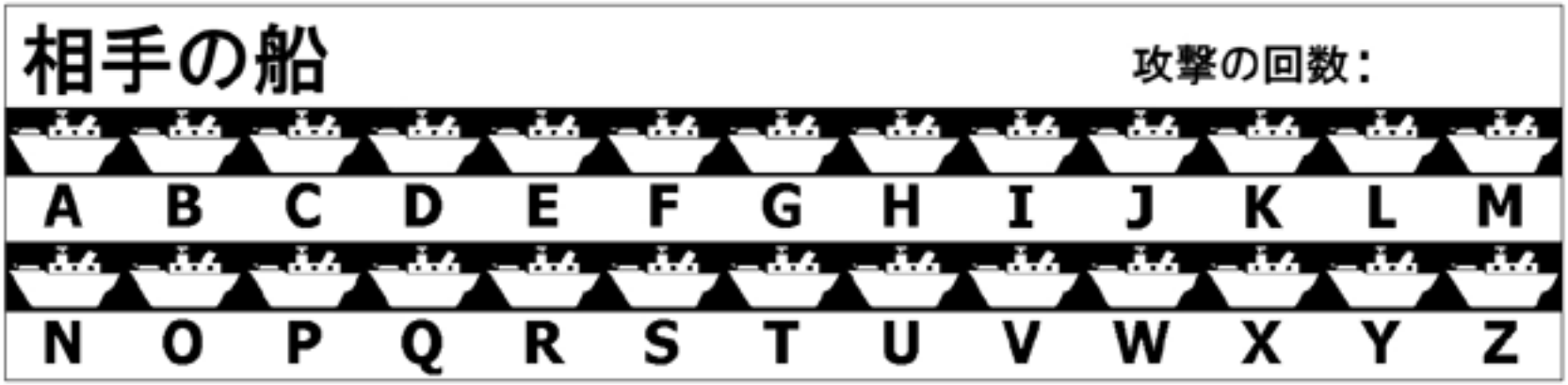
追加学習

① 子どもに3種類のゲームを作らせましょう。2番目のゲームでは、数を小さい順に並べる必要があります。ハッシュ法のゲームでいちばん難しいものを作るやり方を考えさせましょう。（すべての船が1つの列に入るものが最も難しいです）いちばん易しくなる方法も考えさせましょう。（それぞれの列に同じ数だけ船を入れさせてみます）

② もし船が存在しない場合はどうなりますか？（線形探索ではそれがわかるまでに26回攻撃するでしょう。二分探索では5回必要です。ハッシュ法では該当する列にいくつ入っているかで結果が変わります）

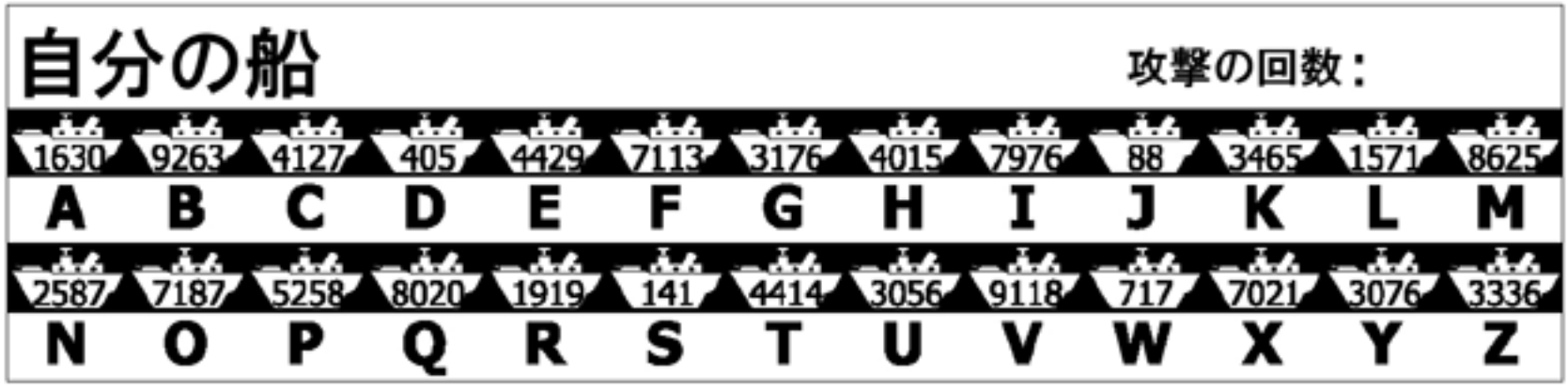
③ 二分探索を使うと、100個のときに何回の攻撃が必要でしょう（約6回）、1000個ならどうでしょう（約9回）、100万個なら（約19回）？（船の数が増えても、攻撃の数はそれほど増えないことに注意しましょう。攻撃は相手が2倍になるごとに1回増えるので、船の数の対数に比例します）

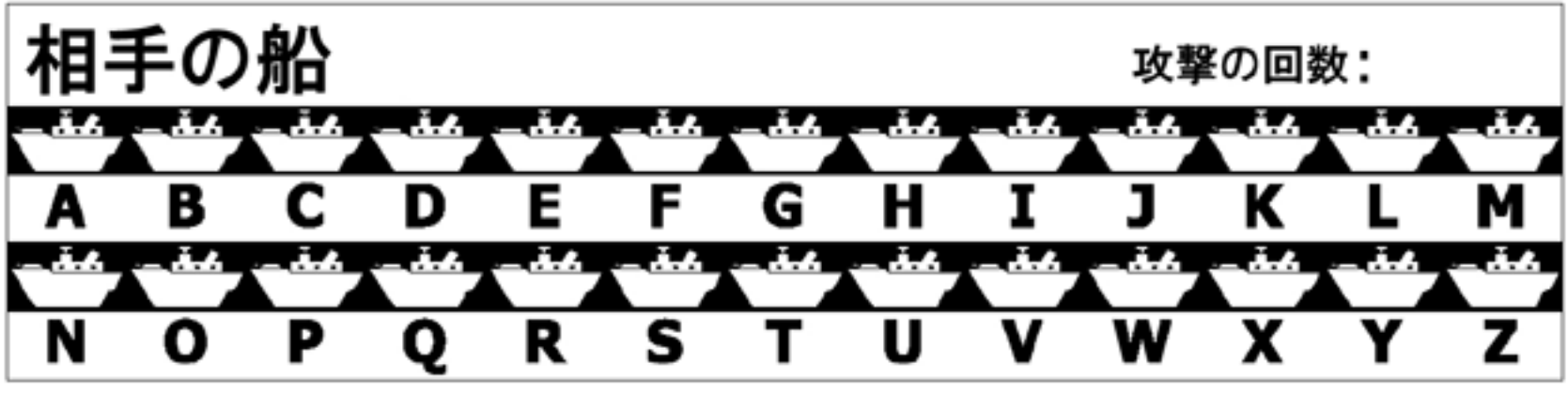

自分の船
攻撃の回数：
9058 7169 3214 5891 4917 2767 4715 674 8088 1790 8949 13 3014
A B C D E F G H I J K L M
8311 7621 3542 9264 450 8562 4191 4932 9462 8423 5063 6221 2244
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

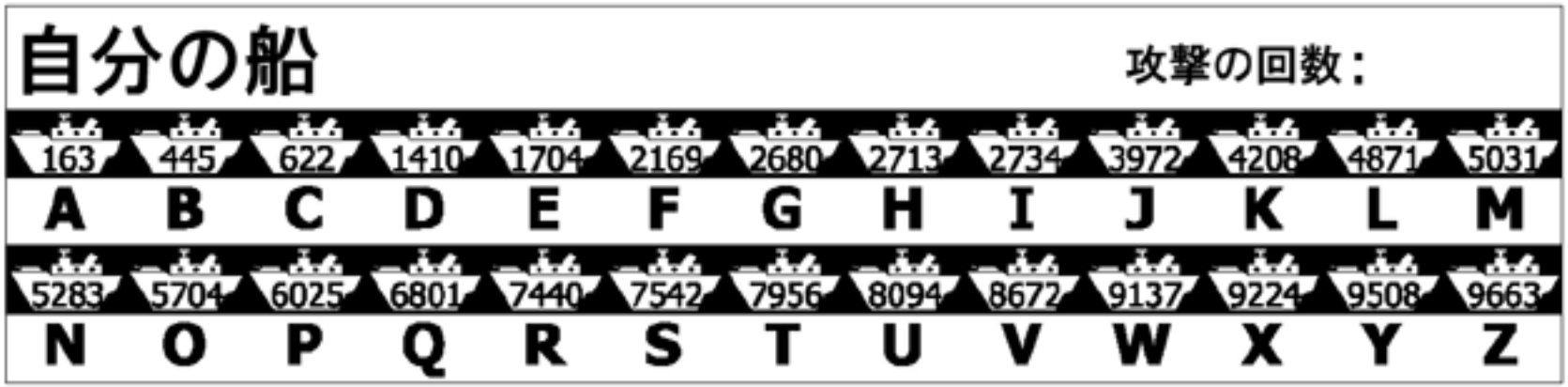
1A

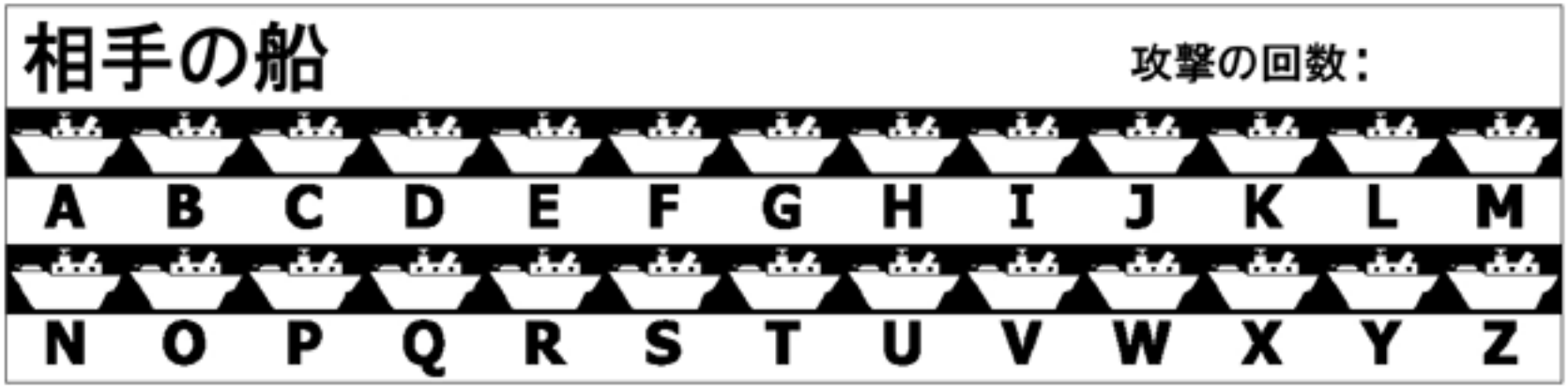
9点


自分の船
攻撃の回数：
1630 9263 4127 405 4429 7113 3176 4015 7976 88 3465 1571 8625
A B C D E F G H I J K L M
2587 7187 5258 8020 1919 141 4414 3056 9118 717 7021 3076 3336
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

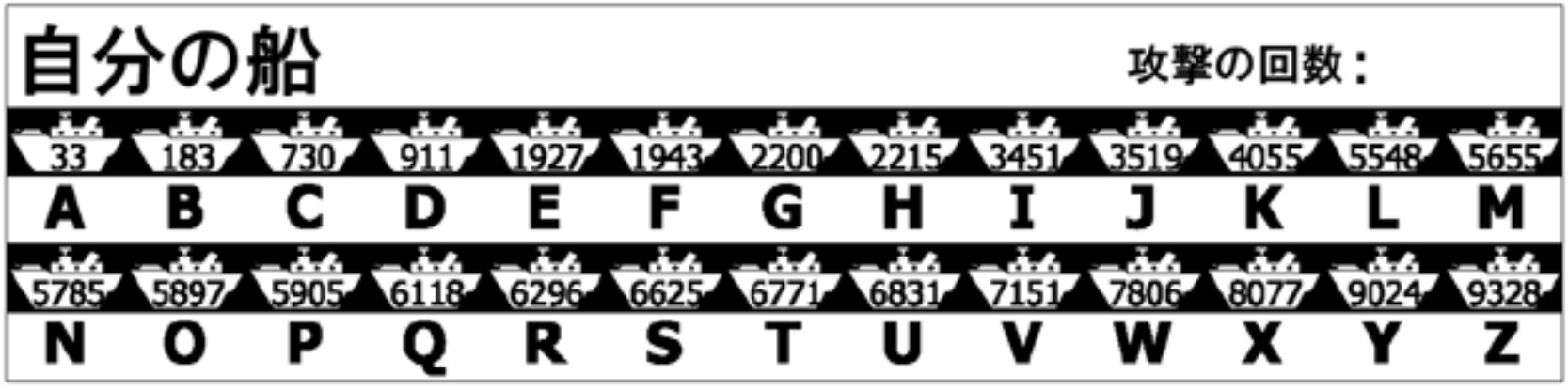
1B

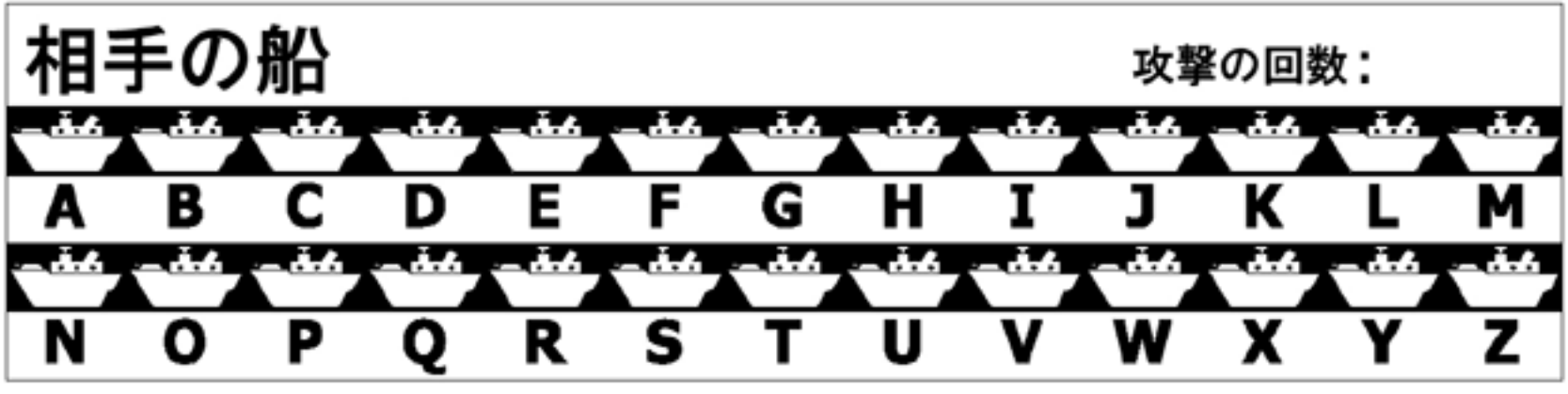

自分の船
攻撃の回数：
163 445 622 1410 1704 2169 2680 2713 2734 3972 4208 4871 5031
A B C D E F G H I J K L M
5283 5704 6025 6801 7440 7542 7956 8094 8672 9137 9224 9508 9663
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

2A

9最点


自分の船
攻撃の回数：
33 183 730 911 1927 1943 2200 2215 3451 3519 4055 5548 5655
A B C D E F G H I J K L M
5785 5897 5905 6118 6296 6625 6771 6831 7151 7806 8077 9024 9328
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

2B

自分の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A 9047	C 3080		E 5125	H 8051	L 7116	O 6000	R 9891	V 4392	W 1062
B 1829	D 9994		F 1480	I 1481	M 8944	P 7432	S 1989		X 2106
			G 8212	J 4712	N 4128	Q 4110	T 2050		Y 5842
				K 6422			U 8199		Z 7057

相手の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A	E	H	K	L		O	R	V	Y
B	F	I		M		P	S	W	Z
C	G	J		N		Q	T	X	
D							U		

3A

自分の船

攻撃の回数：

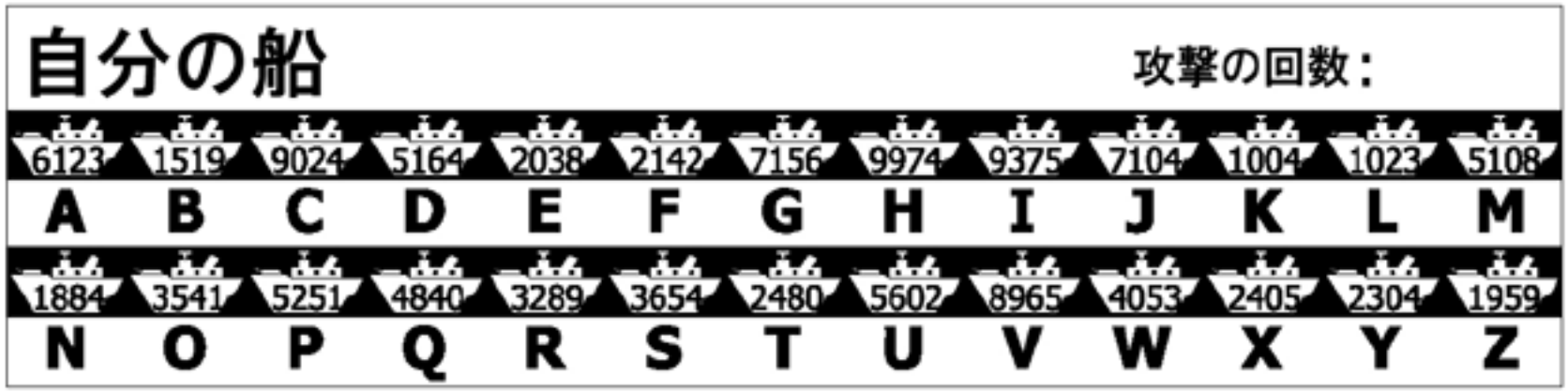
0	1	2	3	4	5	6	7	8	9
A 9308	E 6519	H 1524	K 4135	L 9050		O 4200	R 3121	V 2385	Y 1990
B 1478	F 2469	I 8112		M 1265		P 7153	S 9503	W 5832	Z 2502
C 8417	G 5105	J 2000		N 5711		Q 6028	T 1114	X 1917	
D 9434							U 7019		

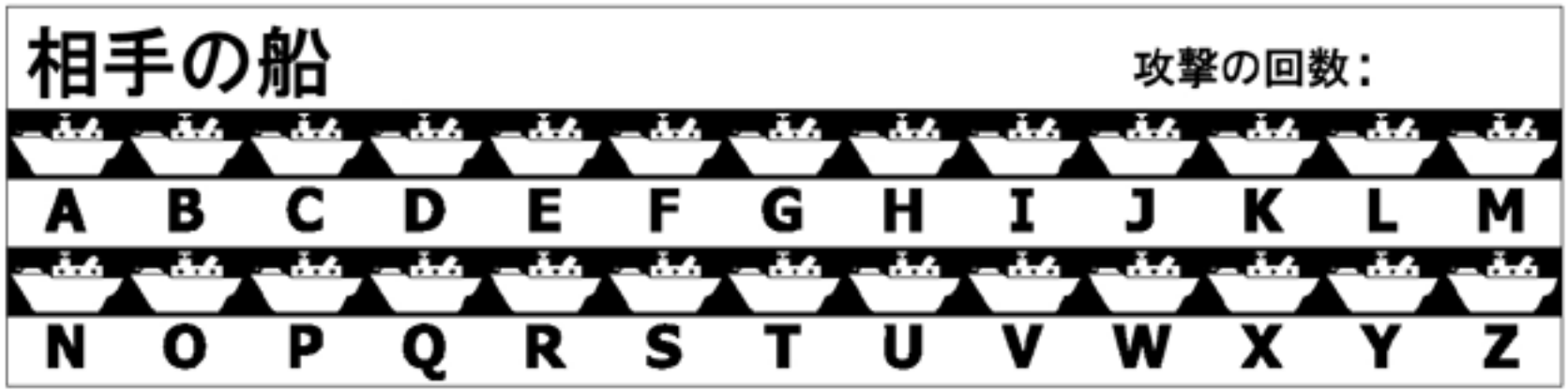
相手の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A	C		E	H	L	O	R	V	W
B	D		F	I	M	P	S		X
			G	J	N	Q	T		Y
				K			U		Z

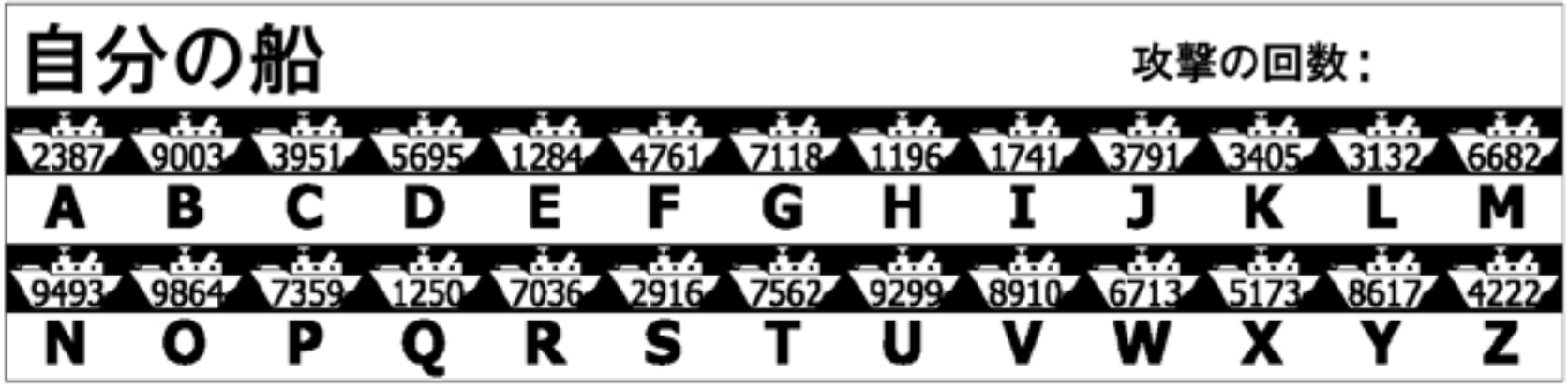
3B

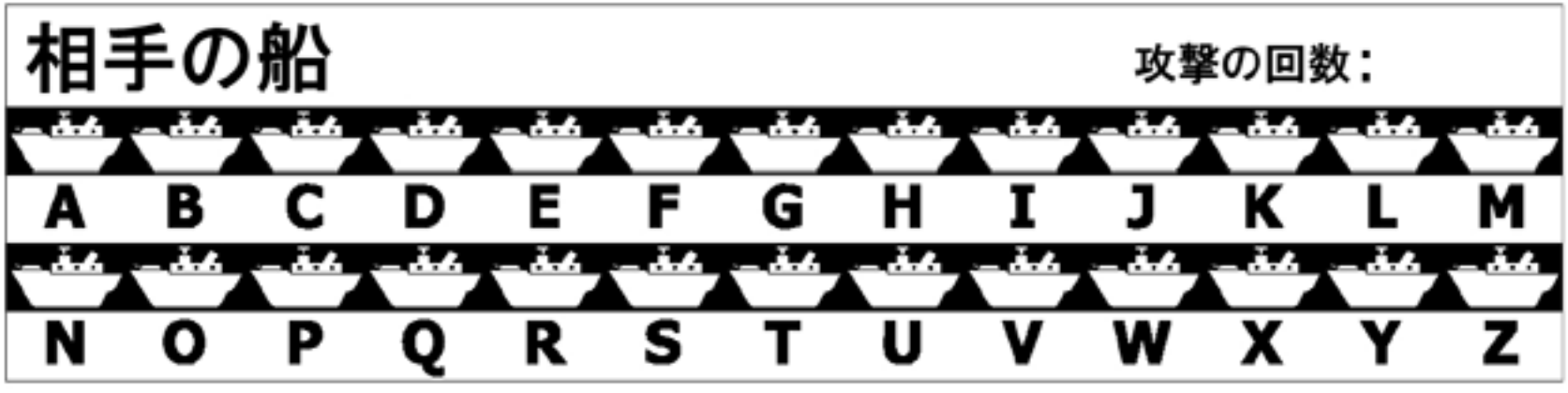

自分の船
攻撃の回数：
6123 1519 9024 5164 2038 2142 7156 9974 9375 7104 1004 1023 5108
A B C D E F G H I J K L M
1884 3541 5251 4840 3289 3654 2480 5602 8965 4053 2405 2304 1959
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

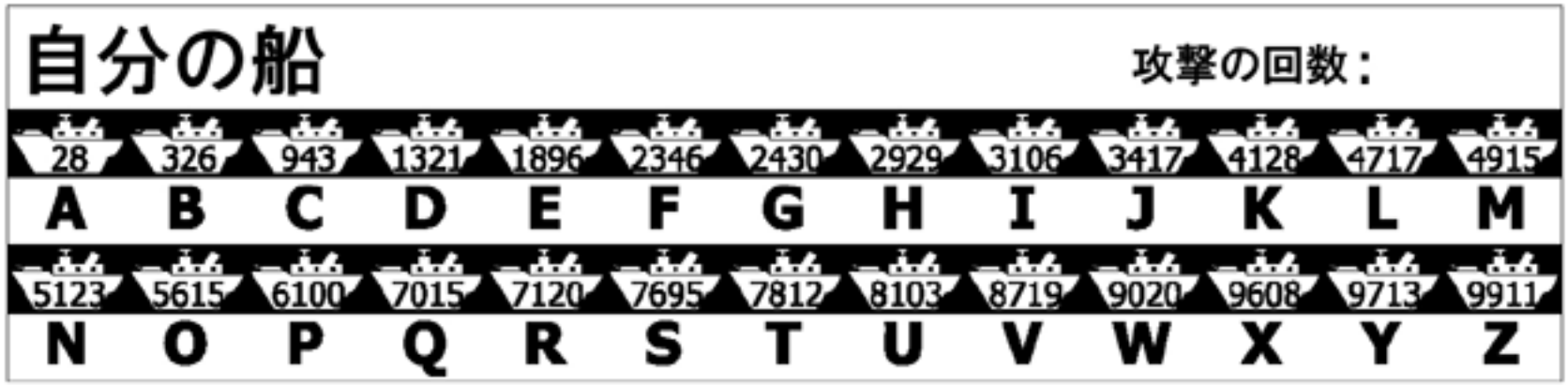
1A′

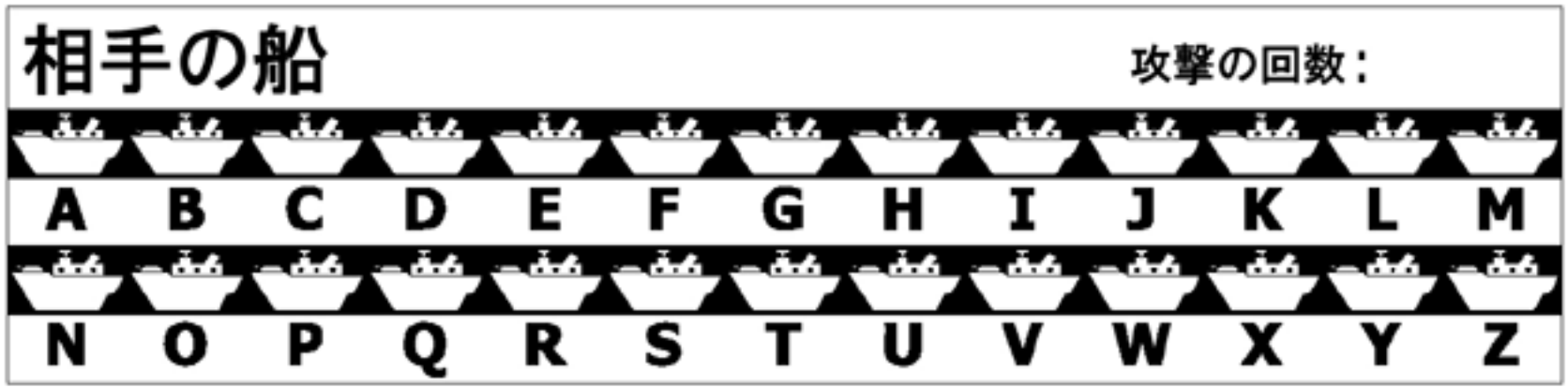
9最点


自分の船
攻撃の回数：
2387 9003 3951 5695 1284 4761 7118 1196 1741 3791 3405 3132 6682
A B C D E F G H I J K L M
9493 9864 7359 1250 7036 2916 7562 9299 8910 6713 5173 8617 4222
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

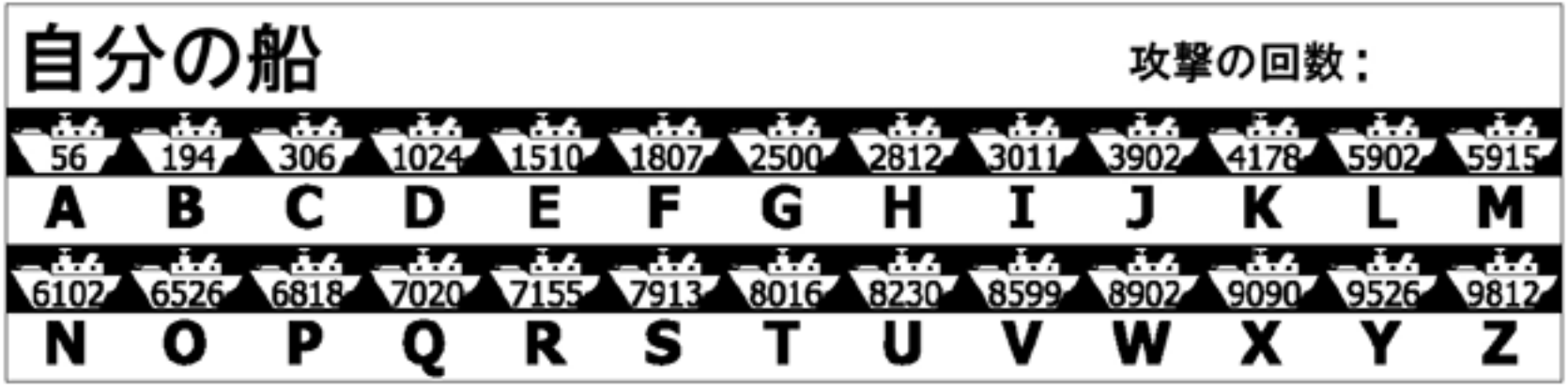
1B′

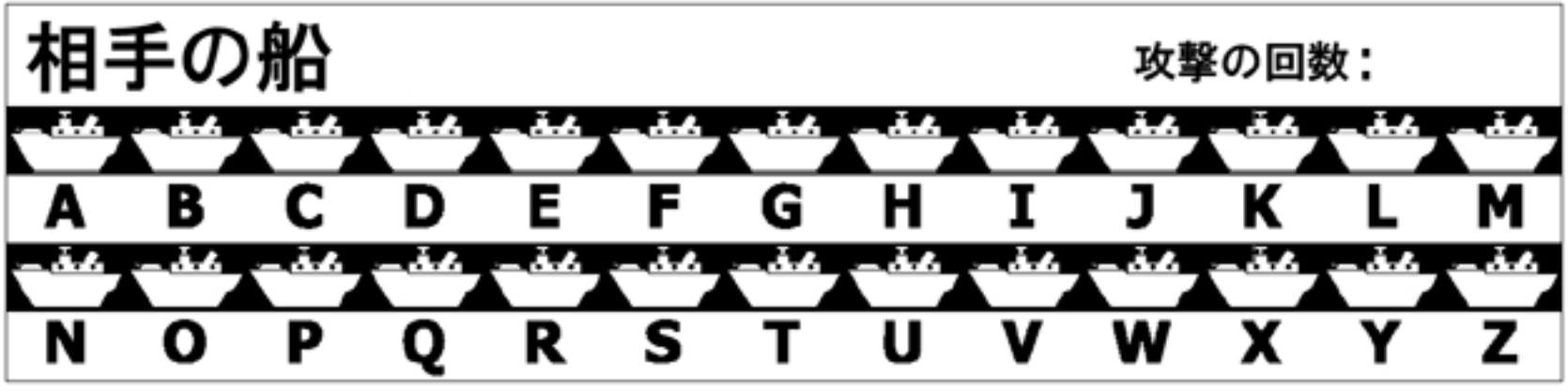

自分の船
攻撃の回数：
28 326 943 1321 1896 2346 2430 2929 3106 3417 4128 4717 4915
A B C D E F G H I J K L M
5123 5615 6100 7015 7120 7695 7812 8103 8719 9020 9608 9713 9911
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

2A′

9最点


自分の船
攻撃の回数：
56 194 306 1024 1510 1807 2500 2812 3011 3902 4178 5902 5915
A B C D E F G H I J K L M
6102 6526 6818 7020 7155 7913 8016 8230 8599 8902 9090 9526 9812
N O P Q R S T U V W X Y Z


相手の船
攻撃の回数：
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

2B′

自分の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A 1982	C 6113		E 9121	H 5009	L 1248	O 2004	R 9369		W 9172
B 7841	D 1055		F 1011	I 2651	M 1716	P 5173	S 1321	V 3285	X 2052
			G 2984	J 1751	N 2148	Q 2806	T 3004		Y 6012
				K 4848			U 7190		Z 7525

相手の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A	E	H	K	L		O	R	V	Y
B	F	I		M		P	S	W	Z
C	G	J		N		Q	T	X	
D							U		

3A′

自分の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A 8615	E 1361	H 7726	K 3000	L 1814		O 9656	R 6993	V 8208	Y 2917
B 7003	F 7644	I 9003		M 2002		P 4002	S 3121	W 9423	Z 4122
C 1991	G 5600	J 5557		N 8844		Q 1221	T 4300	X 4176	
D 6211							U 1907		

相手の船

攻撃の回数：

0	1	2	3	4	5	6	7	8	9
A	C		E	H	L	O	R	V	W
B	D		F	I	M	P	S		X
			G	J	N	Q	T		Y
				K			U		Z

3B′

実際のコンピュータでは

コンピュータは多くの情報を記憶して、それらをすばやく検索できる必要があります。最も大きな課題の１つはインターネットの検索エンジンで、数十億ページの中から数分の１秒で検索することを求められています。コンピュータが検索に使う単語、バーコード番号、著者名などのデータは「検索キー」と呼ばれます。

コンピュータは情報を高速に処理できるので、何か情報を探すときは記憶装置を順に見に行っていると考えるかもしれません。これは線形探索ゲームで体験したやり方です。しかし、このやり方はコンピュータにとってもとても遅い方法です。たとえば、スーパーマーケットの棚に１万種類の商品があることを考えましょう。レジでバーコードを読んだときに、コンピュータは１万個の商品から商品名と値段を探す必要があります。１個のデータを調べるのに 0.001 秒しかかからなかったとしても、商品を全部見るのに 10 秒かかってしまいます。家族のための食料品をレジで支払うのにどれくらい時間がかかるでしょう！

二分探索はよい方法です。この手法では、数は順番に並べられています。真ん中のものを調べることで、どちらの半分に目的のものが入っているかがわかります。この処理を目的のものが見つかるまで続けます。スーパーマーケットの例を考えると、１万個の商品に対して 14 回調べるだけで済み、これは 0.02 秒と、ほとんど気付かない速さで処理できます。

３番目の方法はハッシュ法と呼ばれていました。この方法では、検索キーから情報がある場所がすぐにわかります。たとえば、検索キーが電話番号なら、すべての桁を足して、それを 11 で割った余りを計算しても良いでしょう。データの一部が他のデータから計算されるという意味で、ハッシュキーは「学習４」で扱ったチェックのための数字と似ています。多くの場合、コンピュータは一瞬でデータを探せますが、まれに複数のキーが同じ場所に入ることがあり、その場合には順に探す必要があります。

データを順番に取り出す必要がなく、検索時間の上限を保証する必要がない場合には、コンピュータプログラマはハッシュ法をよく使います。

学習 7 いちばん軽いといちばん重い

整列アルゴリズム

概要

名前を五十音順にしたり、予定や電子メールを日付順にしたり、商品を値段順にするような場合に、コンピュータを使った並びの整列（ソート）が使われます。順序よく並べておくと、そこから物を探しやすくなりますし、値を見やすくなります。もしクラスのテストの点を順番に並べておくと、いちばん小さい点数といちばん大きい点数が一目でわかります。

正しい方法を使わないと、速いコンピュータでも大きい並びを整列するときに時間がかかってしまいます。幸いなことに、高速な整列手法がいくつか知られています。この学習で、子どもは何種類かの整列手法を知り、どのように速く処理できるか良い方法を学びます。

教科学習との関連

・Mathematics: Measurement Level 2 and up. Carrying out practical weighing tasks.

※［量と測定］小３で重さを扱います。

技能

・Using balance scales　天秤を使う。
・Ordering　順番に並べる。
・Comparing　比較する。

年齢

・８歳以上

教材

＜グループごとに必要なもの＞

・大きさは同じだが重さが違う８個の入れ物
　（牛乳パックやフィルムケースに砂を入れたものなど）
・天秤ばかり
・ワークシート（66、67 ページ）

いちばん軽いといちばん重い

話し合おう

コンピュータを使った整列はよく使われます。どんな場面で重要になるかを話し合いましょう。もし整列されていなかったら、どんなことが起きるでしょう？

コンピュータは普通、値を一度に２個ずつ比較します。次のページの学習ではこの制約を使い、子どもに何と似ているかを伝えます。

学習

①　子どもをグループに分けます。

②　グループごとに、66 ページの学習シートと、重りとはかりが必要です。

③　学習の後、結果を話し合わせます。

ワークシート

重りの整列

ねらい

重さがわからないものを重さの順に並べるよい方法を見つける

必要なもの

砂か水、8個の区別できない入れ物、天びん

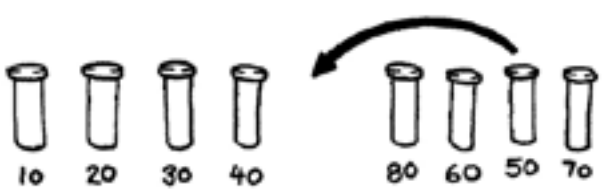
やること

① 入れ物に砂か水を違った量だけ入れ、こぼれないように封をします。
② 順番を入れ替えて、重さの順番がわからないようにします。
③ いちばん軽いものを見つけなさい。簡単に見つけるにはどうしますか？
　注意：入れ物の重さははかりで調べます。いちどに2個の重さを比べられます。
④ 適当に3個の重りを選び、はかりを使って、いちばん軽いものからいちばん重いものまで順に並べなさい。どうやってやりましたか？ 比べる回数を少なくできたのはどんなやり方でしたか？それはなぜですか？
⑤ 次に、重り全体を、いちばん軽いものからいちばん重いものまで順に並べなさい。

整列が終ったら、並んでいる隣同士をもう一度比べて、正しい順に並んでいることを確かめなさい。

選択ソート

選択ソートはコンピュータが使う方法の1つです。どのように動くのかを見てみましょう。最初に、いちばん軽い重りを見つけて、別の場所に移します。これを、重りがなくなるまで繰り返します。



自分が何回重さを比べたかを数えておきましょう。

応用問題

8個を整列するために、何回比較する必要があるかを計算しましょう。
9個ならどうですか？20個なら？

ワークシート

分けて作業する

クイックソート

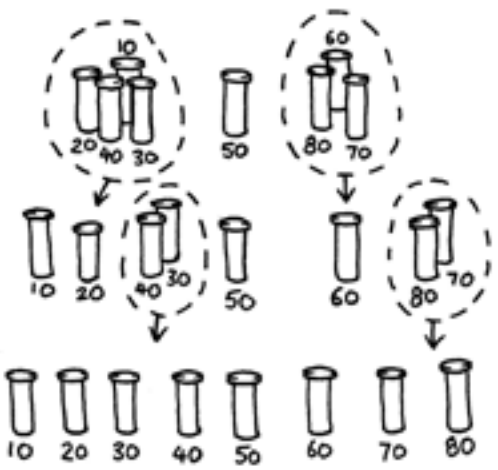
クイックソートは、最も速い手法の1つで、特に並びが大きいときは選択ソートよりずっと高速です。どのように動くかを見てみましょう。

① 適当に1個の重りを選び、天びんの片方に乗せます。

② 次に、残りの重りと比べます。その重りより軽いものは左に、同じものは真ん中に、重いものは右に置きましょう。
（どちらかの数が多くなってしまうことがあるかもしれません）

③ それぞれの組の中で、同じことをやりましょう。
選んだ1個を必ず真ん中に置くのを忘れずに。

これを、それぞれの組の中が1個になるまで繰り返します。最後にすべての組が1個ずつに分けられたら、小さいものから順に並んでいるはずです。



大きさを何回比べる必要がありましたか？

いちばん小さい値やいちばん大きい値を偶然選んでしまわない限り、クイックソートは選択ソートより効率がよいことがわかると思います。選択ソートの比較回数は28回ですが、クイックソートでは、運よく真ん中の値を選べたときは14回で済みます。運が悪くても選択ソートと同じくらいで、ほとんどの場合は選択ソートよりずっと速いです！

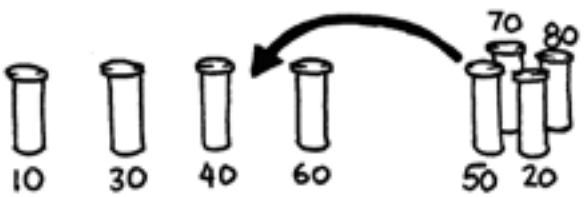
応用問題

常にいちばん小さい値を選んでしまったときは、クイックソートでは何回の比較が必要でしょう？

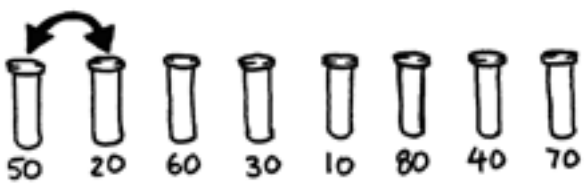
発展と応用

いろいろなソートの方法が発明されています。いくつかを試してみましょう。

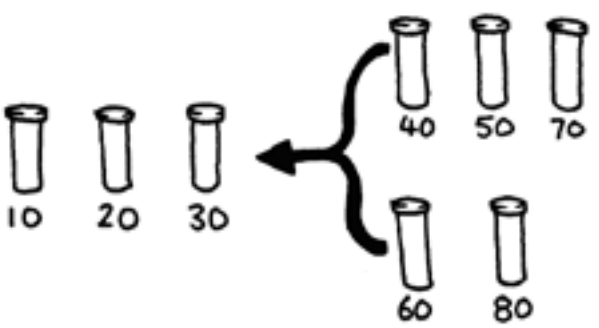
挿入ソートでは、図のように1個ずつ正しい並びの位置に入れていきます。まだ並べられていないものは減っていき、並べられたものが増えていきます。トランプをする人は、よくこの方法でカードを並べます。



バブルソートでは、並びを何度も行き来して、図のように順番が逆のものを交換します。作業は並び全体で交換するものがなくなったときに終わります。この方法はそれほど効率がよくありませんが、理解しやすいと思う人がいるかもしれません。



マージソートは「分けて作業する」方法のひとつです。まず、全体を同じ数になるように2つに分けます（奇数の場合にはどちらかが1個多くなります）。そして、それぞれを整列した後で、2つの並びを統合します。2つの整列された並びを統合するには、並びの先頭を比べて小さいほうを取り出せばよいので簡単です。図では40gと60gが並びの先頭にあるので、40gのほうを取り出して新しい並びに加えています。



実際のコンピュータでは

順に並んでいると情報を探すのが簡単になります。もし電話帳や辞書、本の索引が読みの順に並んでいなかったら、生活はかなり不便になるでしょう。支出額のような数字の並びが整列されていれば、極端な値は最初か最後にあるので見つけるのが簡単になります。同じ値は並ぶので、重複を調べることも簡単です。

コンピュータはたくさんのデータを整列する必要があるため、コンピュータ科学者は効率のよい整列方法を見つけようとしてきました。挿入ソート、選択ソート、バブルソートなどの遅い方法が役立つこともありますが、普通はクイックソートのような高速なものが使われます。

クイックソートでは再帰の考え方が使われています。この考え方では、並びを小さな並びに分けて、それぞれに対して同じように作業を進めます。この方法は「分けて作業する（分割して統治せよ）」と呼ばれます。並びは、作業するために適した大きさになるまで、繰り返し分割されます。クイックソートでは、それぞれの並びは要素が 1 個になるまで分解されます！この考え方は複雑に見えますが、他の方法よりずっと高速です。

解答とヒント

①　いちばん小さな値を見つける最もよい方法は、すべての要素を見て、それまでのいちばん小さな値を覚えておくことです。そのために、２つの値を比較して、小さなほうの値を覚えます。次に、もう１つと比較します。そしてすべての要素を調べます。

②　天秤で３個の重さを比べましょう。３回の比較が必要ですが、子どもが推移則に気づいたときは２回で済むことがあります。（A が B より小さく B が C より小さいなら、A は C より小さいことがわかります）

上級者向け

選択ソートで比較する回数を楽に計算する方法を紹介します。

２個の要素から小さいほうを見つけるために、１回の比較が必要です。３個では２回、４個では３回が必要です。８個の要素を考えると、最初の１個を見つけるためにに７回の比較を行い、次の１個のために６回の比較を行い、次に５回の比較、というように、次の比較が必要です。

7 + 6 + 5 + 4 + 3 + 2 + 1 = 28 回の比較。

n 個の要素を整列するためには、1 + 2 + 3 + 4 +... + n − 1 回の比較が必要です。これらの和は、順序を変えることで簡単に計算できます。

たとえば 1 + 2 + 3 + ... + 20 の和は、次のように数の組み合わせを作れます。

□1 + 20□ + □2 + 19□ + □3 + 18□ + □4 + 17□ + □5 + 16□ +
□6 + 15□ + □7 + 14□ + □8 + 13□ + □9 + 12□ + □10 + 11□
= 21 × 10
= 210

一般に、和は次のように計算できます。1 + 2 + 3 + 4 ... + n − 1 = n□n − 1□/2

学習８　時間内に仕事を終えろ

並び替えネットワーク

概要

コンピュータがいくら速くても、問題を解決できる速さには限界があります。スピードを上げる１つの方法として、問題をいくつかに分けて複数のコンピュータで処理する方法があります。この章では、同時に複数の比較を行う並び替えネットワークを扱います。

教科学習との関連

・Mathematics: Number level 2 and up. Exploring number: Greater than, less than

※［数と計算］数の比較：より大きい、以上、以下、未満、範囲。数の大小に関しては小１から扱っていますが、学習指導要領で「≧」は、数と式の領域で中１に配当されています。

技能

・Comparing　比較する。
・Ordering　順序付ける。
・Developing algorithms　アルゴリズムを開発する。
・Co-operative problem solving　協調的な問題解決を行う。

年齢

・７歳以上

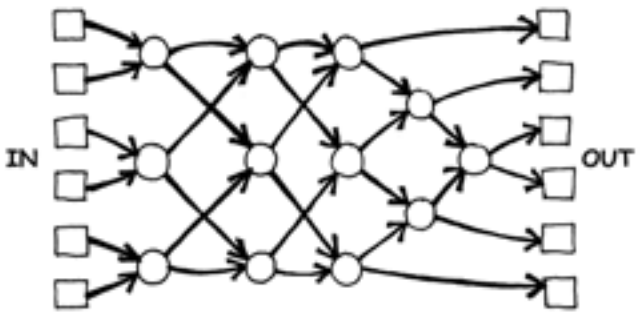
教材

＜外でのグループ活動に必要なもの＞

・チョーク
・６枚のカードが２組 複写用シートを切ってカードに貼ります
・ストップウォッチ

並び替えネットワーク

まず、地面に子どもが歩けるくらいの大きさでこのネットワーク図を描きます。教室では、大きな紙かシートを使うとよいでしょう。



指示

この活動では、並べ替えネットワークを使ってコンピュータがでたらめに並んだ数字をどのように小さい順に並べ替えるかを学びます。

①　6人のグループを作りましょう。1度に1個のグループしかネットワークを使えません。

②　グループのメンバーは1人1枚ずつ、数字が書かれたカードを持ちます。

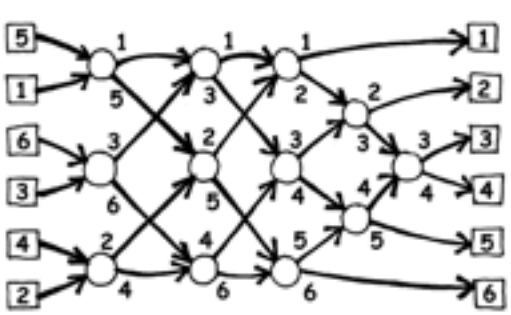
③　グループのメンバーは、ネットワークの描かれたコートの左側に立ちます。カードの並びはランダムです。

④　線に沿って歩きます。○のところで、他の友達が来るまで待ちます。

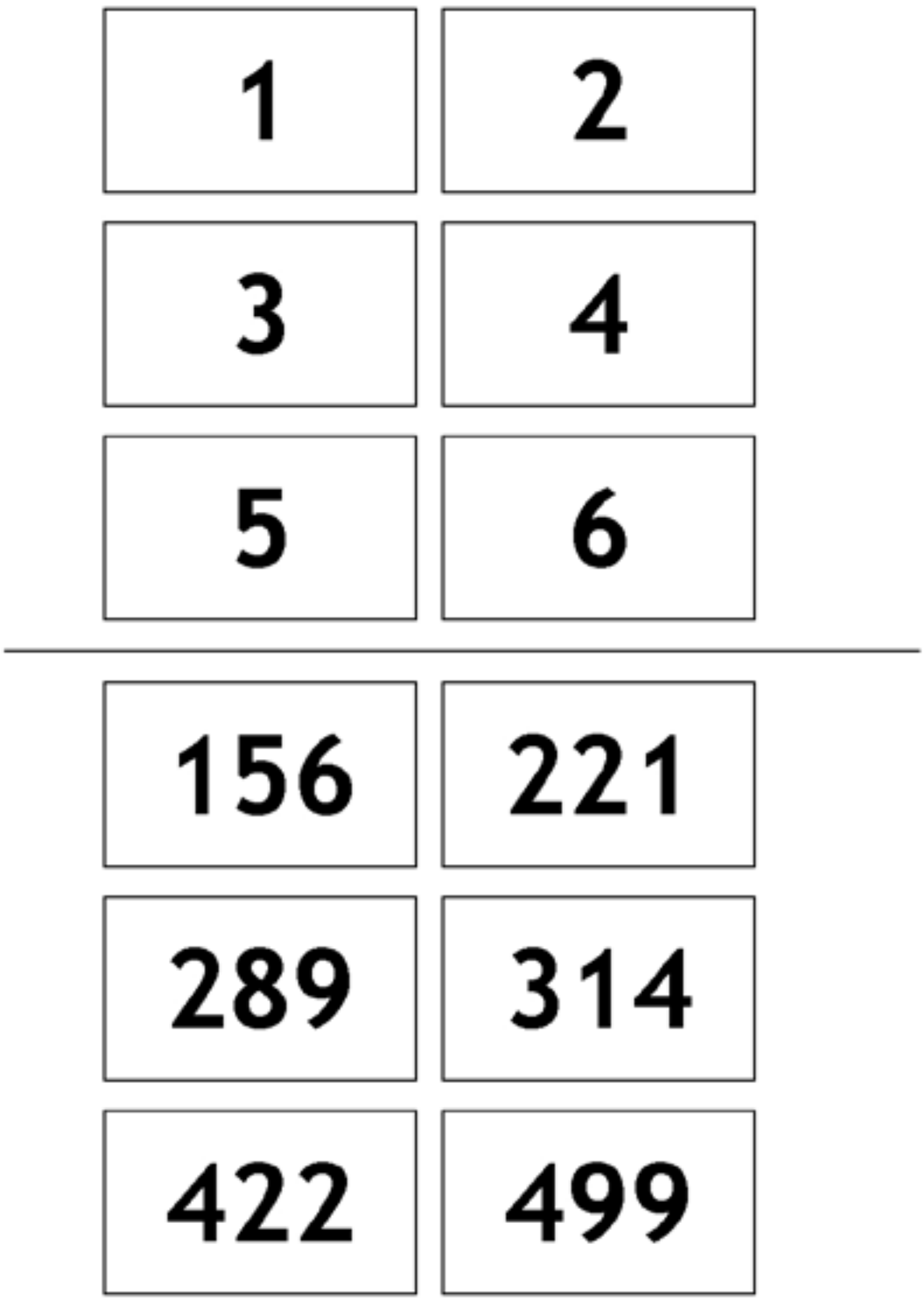
⑤　友達が○のところまできたら、あなたのカードと友達のカードを比べてみましょう。小さい方のカードを持っている人は、左の線にそって進みます。大きい方のカードを持っている人は右の線に沿って進みます。

⑥　端までいったとき、あなたは正しい位置にいますか。

間違ったらもう一度最初からやり直します。ネットワークの接点での操作、例えば小さい方が左へ、それ以外は右へ流れるということを理解できているか確認しましょう



複写用原本：並び替えネットワーク



バリエーション

① 子どもがこの活動に慣れてきたら、ストップウォッチを使って各グループがネットワークを抜けるのにどれくらい時間がかかったかを計ってみましょう。

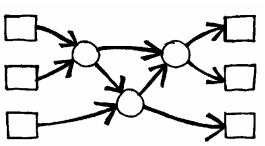
② 大きい数字のカードを使いましょう。(3桁のものをコピーして使ってください)

③ 比較するのにちょっと大変な大きめの数字のカードを作ってみましょう。または、言葉のカードを使ってABC順に並べることもやってみましょう。

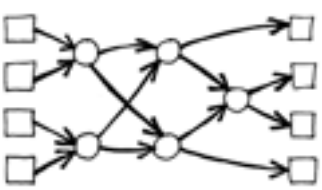
発展学習

① 小さい数字を持った人が左に進む代わりに右へ行ったらどうなるでしょう。逆も同じかな？(数字はおそらく、逆の順に並ぶでしょう。)
ネットワークを逆方向に進むとどうなりますか？(必ずしもうまくは行きません。子どもは逆に入れたときに正しく出てこない例があることに気づくでしょう)

② もっと小さいネットワークや、もっと大きいネットワークを作ってみましょう。これは3個の数字を並べ替える例です。子どもは、自分で作ってみるべきです。

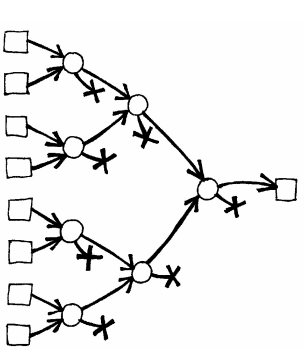


③ 下の図は4つのデータを並び替える2つのネットワークです。どちらが速いでしょうか。(2番目のものです。1番目は、ひとつひとつ順番にすべてのものを比較しますが、2番目のものは、同時に行われています。最初のものは逐次処理の例ですが、2番目のものはそれより速く動作する並列処理の例です。)



④ もっと大きな並べ替えネットワークを作りましょう。

⑤ ネットワークは、入力データのもっとも小さな値やもっとも大きな値を見つけることにも使われます。これは8個のデータのネットワークで、入力データの一番小さな値を出力します。(他のデータはネットワーク内で消えてしまいます)



⑥ 日常生活で並列処理が役立っていることを考えましょう。たとえば、料理のときに道具を1個しか使わないとすごく時間がかかります。たくさんの人を雇えば早く終わる仕事は何でしょう？また、そうでない仕事は？

実際のコンピュータでは

コンピュータが多く使われるにつれて、情報を高速に処理する必要性が高まっています。コンピュータのスピードをあげるために、コンピュータに処理させる命令数が少なくて済むプログラムを書く方法があります。（学習６と７で説明しました）

問題をより速く解くためのその他の方法として、仕事をいくつかに分けて複数のコンピュータで同時に処理する方法があります。例えば、６個の数字を並べ替える場合、全部で 12 回、数字を比較しますが、３つの比較は同時に行えます。これは、５つを比較するのに必要な時間でできてしまうということです。この並べ替えネットワークは、いちどに１個ずつの比較を行う場合よりも２倍以上速くなります。

並列処理を行えばすべての仕事が早く終わるわけではありません。アナログ的な例ですが、10 メートルの溝を掘っている人を想像してください。もし 10 人が手分けして１メートルずつ掘ったら、その仕事はもっと早く終わるでしょう。しかし、10 メートルの深さの穴を掘る場合には同じというわけにはいきません。最初の１メートルが掘り終わるまで、２番目の人は掘れないのです。コンピュータ科学者は、コンピュータを並列に働かせることで高速に問題を解決するための方法を見つけようと努力しています。

学習９　マッディ市プロジェクト

最小全域木

概要

私たちの社会は、電話、エネルギー供給、コンピュータ、道路などの様々なネットワークで結ばれています。それぞれのネットワークでは、道路、ケーブル、無線などをどのように配置するかという選択が行なわれています。効率よくネットワークを組む方法を知ることは大切です。

教科学習との関連

・Mathematics: Geometry Level 2/3 and up. Exploring shape and space: Finding the

shortest paths around a map

※［量と測定］小３で学習する「キロメートル」に関連させて、身の回りの地域の地図を読んだり、最短の経路を考えたりします。

技能

・Problem solving　問題解決を行う。

年齢

・９歳以上

教材

＜子どもごとに必要なもの＞

・ワークシート：マッディー市プロジェクト（ 78 ページ）

・おはじき、あるいは、厚紙を小さな四角に切ったもの（だいたい 1 人当たり 40 個）

マッディ市プロジェクト

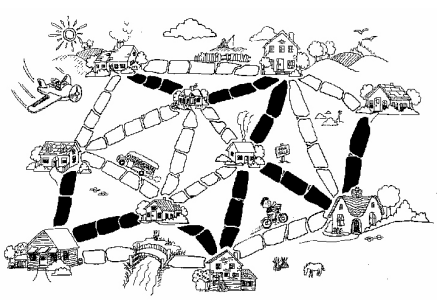
導入

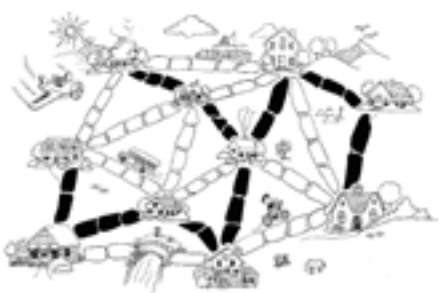
この活動では、コンピュータが現実世界の中でどのように利用されているかを見ることができます。電気を供給する電線や、都市ガスを供給するガス管などをすべての家庭に引きこむのに、最善の方法を見つけるといったものを扱います。子どもに 78 ページのワークシートを配りましょう。これは「マッディー市プロジェクト」を説明するものです。

補足的な話し合い

子どもが見つけた解き方を発表させて、たがいに参考にさせます。どのような作戦を用いたのでしょうか。

最善の方法を見つける方法のうちの１つは、線が書き入れられていない「白地図」の上に「おはじき」を置くことで、経路工事したことを表すことです。家から家へ経路を工事して、すべての家をネットワークに結びつけるまで、段々とおはじきを置いていきます。工事することで、経路の長さは増えて行きます。しかし、すでにネットワークにつながっている家をつなげる必要はありません。経路をつなげる順番を変えれば、異なる解が見つかります。２つの解を下に示します。





もう１つの方法は、全部の道におはじきを置いてから、不要なおはじきを取り除いて行く方法です。こちらの方が、時間がかかります。

現実世界でネットワークはどこで使われているでしょう。

このようなネットワークを表したものを、コンピュータ科学者は「グラフ」と呼びます。現実世界のネットワークをグラフに表現して、最善のネットワークを作るといった問題解決を行っているのです。現実世界のネットワークには、町と町とを結ぶ道路、あるいは、地域と地域とを結ぶ航空路などが考えられます。

グラフを扱う方法（アルゴリズム）には、まだまだ多くの方法があります。たとえば、点（都市）から点までを最短の距離で結ぶ方法や、すべての点を巡るのに最短の経路を見つける方法などがあります。

ワークシート

マッディー市プロジェクト

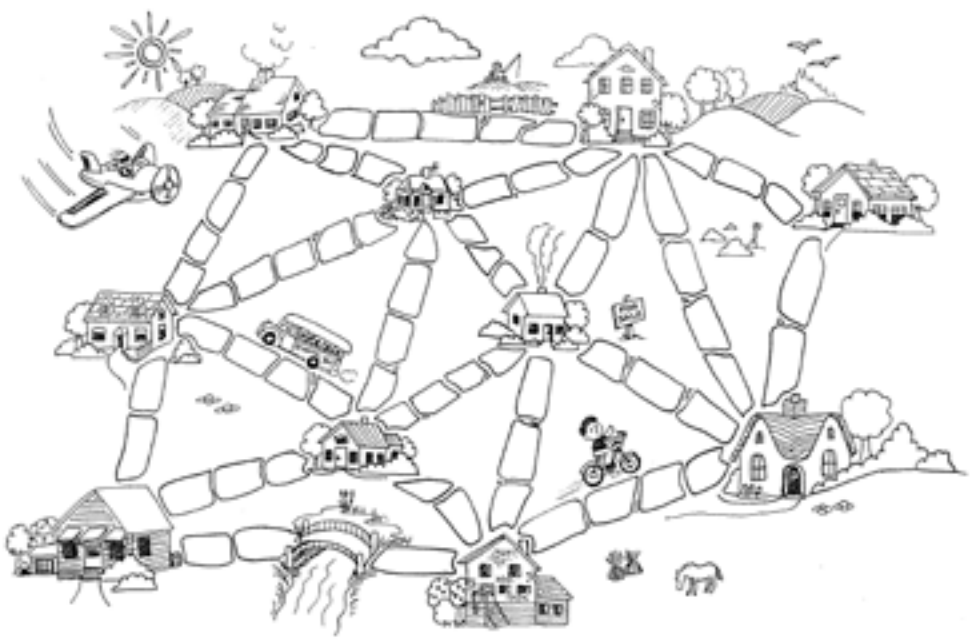
昔、まともな道路が1本もない町がありました。雨のあとなど、この町を歩いて回るのは大変でした。地面がぬかるんでいるので、車は泥に車輪をとられて立ち往生し、靴は泥で汚れてしまうからです。この町のことを、人々は英語でマッディー市、つまり「泥の町」と呼びました。

この町の市長は一部の道を舗装することを決意しましたが、必要以上のお金を使いたくありませんでした。それは市民プールも欲しいと思っていたからです。そこで、市長は2つの条件を決めました。

① 舗装された道を通って、どの家からどの家にも行けるような舗装をしなければならない。
② 舗装の費用はできるだけ安くする。

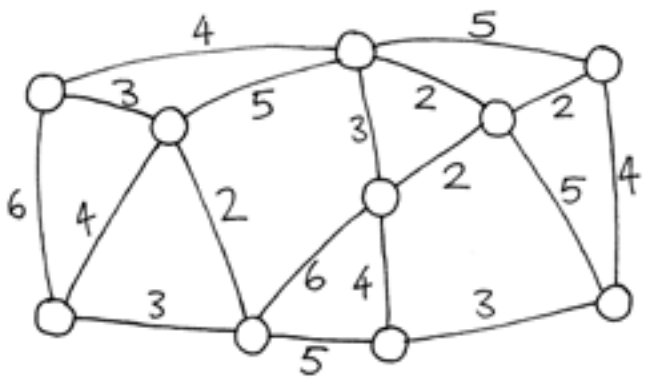
ここに町の地図を示します。家と家との間の敷石の数は、そこを舗装したら費用がいくらかるかを表しています。全ての家を舗装された道につなぎ、しかも費用ができるだけ安くなる経路を見つけましょう。（橋は敷石に数えません）

どんな方法で経路を見つけましたか？



発展と応用

都市と道路を表現するのに次のような方法もあります。



家（節点）は円で、泥道（経路）は線で、そして泥道の長さは線の横に書かれた数で表されています。

コンピュータ科学者や数学者は、この種の図式を「グラフ」と呼び、このような問題を表すのによく使います。「棒グラフ」や、「折れ線グラフ」でも「グラフ」という言葉を使うのでまぎらわしいのですが、コンピュータ科学者が、「折れ線グラフ」の意味で「グラフ」という言葉を使うことはめったにありません。経路の長さは線の横の数で表しているので、線の長さを経路の長さに一致させる必要はありません。

「マッディー市」のような未舗装の道を最小費用で舗装する問題を作ってみましょう。地図はグラフの形で示せばよいです。そして友だちにその問題を解かせてみましょう。

最善の方法では何本の道の舗装が必要になるかという規則を見つけられますか？
それは、家の数と関係していますか？

実際のコンピュータでは

電気、ガス、水道などを新しい住宅地に供給する方法を設計することを考えましょう。電線や管を会社からすべての家庭につなげる必要があります。それぞれの家は何らかの形でネットワークにつながっている必要がありますが、どの経路をとるかは問題ではありません。

ネットワークの経路の全長が最小になるようデザインする問題は、「最小全域木問題」と呼ばれています。

この「最小全域木問題」は、ガスや電気のネットワークにとどまらず、コンピュータ、電話、石油、航空路のネットワークなどに役立っています。しかし、旅行をするときに、旅行者にとって最善のルートを決めるとなると、費用がどれだけかかるかと同様に、どれだけ便利かということも考慮しなければなりません。いくら安いからといって、次の都市へ行くのに何時間も飛行機に押し込められるのは誰も望まないでしょう。ですから、ここで考えた「最小全域木問題」のアルゴリズムは、旅行のプランの問題にはそれほど使われません。単に経路、つまり航空路の「全長」を最小にするものだからです。

最小全域木は、グラフに関するほかの問題を解くときの考え方のひとつとしても役立ちます。例えば「巡回セールスマン問題」は、そのような問題で、ネットワーク上のすべての節点を最小の経路で訪れる方法を見つけようとするものです。

最小全域木問題を解くのに効果的なアルゴリズムは、いくつか知られています。単純な方法としては、クラスカルのアルゴリズム（J.B. クラスカルが、1956 年に発表しました）があります。何も経路のない節点だけの状態から最適な解を見つける方法で、前にネットワークにつながっていなかった部分への経路を加えることのみによって、ネットワークを大きくして行くものです。

「巡回セールスマン問題」など、グラフに関する多くの問題に関しては、今もコンピュータ科学者は最善の解を十分速い速度で見つける方法を発見しようとしています。

解答とヒント

発展と応用（79 ページ）

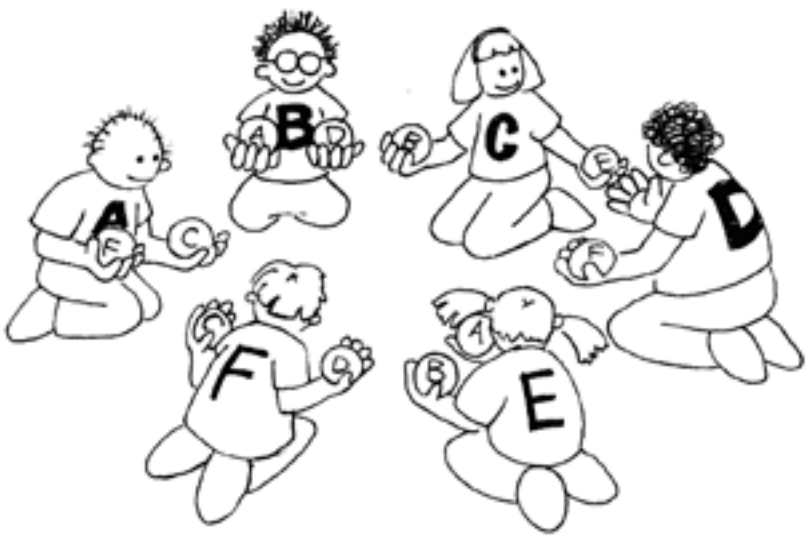
町に n 軒の家がある場合、何本の道（経路）が必要だろうか？

最適な解はいつもちょうど n-1 本の経路を持つことがわかる。

n 軒の家をつなげるのにこれだけの本数があれば十分であるから、1 つ家を加えると 1 本これまで余計だった経路をつけ加えることになる。

学習 10　みかんゲーム

ネットワークにおけるルーティングとデッドロック



概要

たくさんの車が一本の道路を走ったり、インターネット経由で多くのメッセージをやりとりするなど、1 つの資源を大勢の人々が利用するときは、互いに手詰まりになる「デッドロック」の状態になる可能性があります。これを避けるためには、協調的にものごとを進める必要があります。

教科学習との関連

・Mathematics: Developing logic and reasoning

※数学 : 論理と推論能力の育成

技能

・Co-operative problem solving　協調的な問題解決を行う。

・Logical reasoning　論理的推論を行う。

年齢

・9 歳以上

教材

＜子どもごとに必要なもの＞

・みかんまたはピンポン球 2 個

・名札または名前シール

みかんゲーム

導入

これは、協力して行う問題解決のゲームです。参加者全員が、自分の名前が書かれたみかんを持って終わることが目標です。

① 5 人以上の子どもでグループを作り、それぞれが丸く向き合って座ります。
② 子どもは 1 文字の名前を持ちます（名札やシールに書いておきましょう）。子どもの文字を書いたみかんを 2 個ずつ用意します。ただし、1 人はみかんを 1 個にして、いつも誰かの片手があいているようにします。
③ 子どもに、みかんをランダムに配ります。1 人を除き、全員に 2 つのみかんを持たせます。その結果、1 つだけ持つ子どもが 1 人います。（最初はどの子も自分の文字が書かれたみかんを持たないようにします）
④ 子どもは隣同士でみかんを手渡し、自分と同じアルファベットが書かれたみかんを全員が持つまでそれを続けます。手渡すときのルールは以下の 2 つです。
   a）1 回に相手に手渡すみかんは 1 つだけです。
   b）すぐ隣の人の手が空のときだけ、みかんを手渡すことができます。
   （つまり、隣の人にみかんを手渡すことができる子どもは 1 回に 1 人だけです。）

子どもはすぐに、「自分がみかんを手に入れたら他の人にみかんを渡さなくなる」ような欲張りでいると、ゲームが終わらなくなることに気づきます。このとき、だれかがゲームで「勝つ」のが目的ではなく、全員が自分のみかんを持ったときパズルが解けるということを話してきかせる必要があるかもしれません。

振り返りの議論

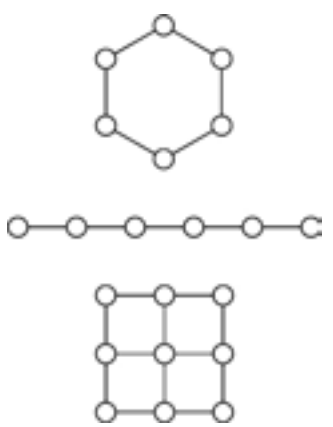
子どもは問題を解決するために、どのようなアイデアを用いましたか？

実生活ではどのような場所でデッドロックを体験するでしょうか？（交通渋滞、野球の塁と走者、大勢が出口に向かう様子など）

発展学習

グループの人数を変えて試してみましょう。

- 子どもに新しいルールを考えさせてみる。
- 話し合いをしないでやらせてみる。
- 他の形を試す。
  一直線に並ぶ
  一部の子どもは 2 人以上の隣がいるようにするなど



実際のコンピュータでは

ルーティングは経路制御と呼ばれ、物や情報がどのように経路を伝わるかを取り決める規則です。デッドロックは、互いに必要な資源を持っていることが原因で、それ以上処理を進められなくなった状態です。輻輳（ふくそう）は、経路が混み合ってしまい、ほとんど物や情報が流れなくなった状態です。

これらのルーティング、デッドロック、輻輳は、さまざまなネットワークでいらだたしい問題を引き起こすことがあります。ラッシュアワーの道路を思い浮かべてください。東京やニューヨークでは、道路がまさにデッドロック状態のように大混雑して、誰も自分の車を動かすことができないようなことが何度も生じてきました。また、銀行などの業務で使うコンピュータが「ダウンする」問題が、通信ネットワークのデッドロックが原因であったこともあります。ネットワークを、ルーティングが容易で効率的であり、輻輳が最小限になるように設計することは多くの分野のエンジニアが直面する難しい問題です。

複数の人が同じ 1 つのデータを要求するのはよくあることです。顧客の預金残高のようなデータを更新するときには、その間に「ロック」をかけて他人に見えないようにしておくことが重要です。ロックされていないと、別の誰かがそのデータを同時に更新してしまい、間違った預金残高が記録されてしまうことにもなりかねないからです。しかし、こうしたロックの設定が他の項目のロック設定によって妨げられる状況が発生すると、デッドロックが起こります。

コンピュータ設計の最も画期的な成果の 1 つは、並列計算機の登場です。これは、数百台から数千台ものパーソナルコンピュータのようなプロセッサをネットワークで結合し、1 台の強力なコンピュータとしたものです。このような並列計算機をうまく動作させるために、みかんゲームのような多数の問題が内部のネットワークで常に（人間が行うのよりも、もっとずっと高速に）実行されているのです。

第３部
コンピュータに何をすべきか教える

手続きの表現

コンピュータに何をすべきか教える

コンピュータは１秒間に何百万もの命令に従って動いています。コンピュータに何をすべきかを教えるには正しい命令を与えればいいのですが、それは言うほど簡単ではありません。

私たちは、与えられた命令の意味を、常識を働かせて解釈します。誰かが「ドアを通ってきて」と言ったら、本当にドアを蹴破ってきなさいという意味ではなく、「(必要であればドアを開けて)通ってきなさい」という意味になります。しかし、コンピュータは違います。実際に、コンピュータが移動ロボットに取りつけられたなら、人間はコンピュータがその言葉通りにドアを通り抜けようと解釈して動作することで、損傷や危険なことが起きないよう安全に十分注意する必要があります。命令に対して正直に「何も考えずに」従うようなものを取り扱うことには多少の慣れが必要です。

本節の２つの学習では、限られた命令を使って、文字の通りに動く機械とやり取りするための考え方を扱います。

最初の学習では、コンピュータが取り扱うことのできる記号、すなわち単語、数値、文字列を認識するためにコンピュータが使う「有限状態オートマトン」と呼ばれる仕組みについて学びます。

２番目の学習では、人間がコンピュータとやりとりする方法について学びます。良いプログラマになるためには、コンピュータが解釈する命令の組をどのように用いてコンピュータに伝えるかを理解する必要があります。そうした命令の並びがプログラムです。こうした命令を記述するためにプログラミング言語にはさまざまなものがありますが、本書ではコンピュータなしでも利用できるシンプルな言語を用います。

学習 11　宝探し

有限状態オートマトン

概要

コンピュータのプログラムでは、記号の並びを処理することがよくあります。記号というのは、文書の中の文字や単語、あるいはプログラムの文などです。たとえば「整数は 0 から 9 の数字が並んだもの」「実数は数字の並びの後に 1 個の小数点があり、その後に数字が並んだもの」という規則を表すには、この学習で扱う有限状態オートマトンの考え方を使います。この章では、コンピュータが単語や文字列を認識するときに相当する図として、宝の地図を使った作業を行います。

教科学習との関連

・Mathematics: Developing logic and reasoning—using words and symbols to describe
and continue patterns
※数学：論理と推論能力の育成言葉や記号を用いてパターンを記述したり、パターンの続きを作ったりする。
・Social Studies　社会科
・English　国語

技能

・Simple map reading　簡単な地図を読み取る。
・Recognising patterns　パターンを認識する。
・Logic　論理を理解する。
・Following instructions　命令を理解する。

年齢

・9 歳以上

教材

＜教師が用意するもの＞
・一組の地図カード（最初に見せるとき、命令部分が地図を描く子どもから見えないように注意すること）
・複写用シートの島のカード（93 ページ以後に掲載）を複写し切り抜く。 点線に沿って折り、カードの正面が島の名前で背面が命令となるように糊付けする。
＜子どもごとに必要なもの＞
・ワークシート：宝島の財宝への行き方を見つけよう（ 92 ページ）
・ペンまたは鉛筆
＜選択の発展学習のために子どもごとに必要なもの＞
・ワークシート：宝島（98 ページ）
・ワークシート：謎のコインゲーム（99 ページ）

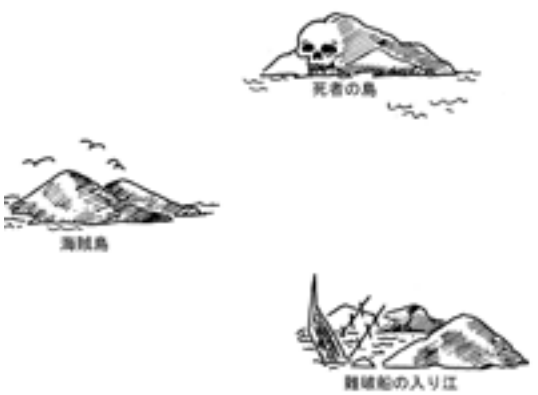
宝島

導入

宝島を見つけましょう。海賊船の仲間たちが島々の間を旅人を乗せて決められた航路を通って航海します。宝島へ行く最も良い航路を見つけてください。島に到着すると、A、B の船のどちらに乗るかを聞かれます（両方は選べません）。島にいる人から、その船が次にどの島へ行くのかを教えてもらえますが､海賊たちは全部の島の地図は持っていません。手元の地図を使って、どの島を通ってどちらの船に乗ってきたのか記録していきましょう。

実演

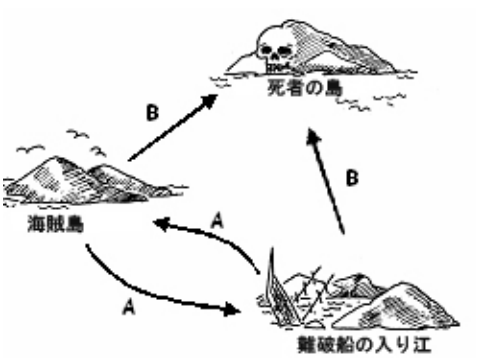
スライドまたは黒板に、右のような３つの島からなる図を描いてください。



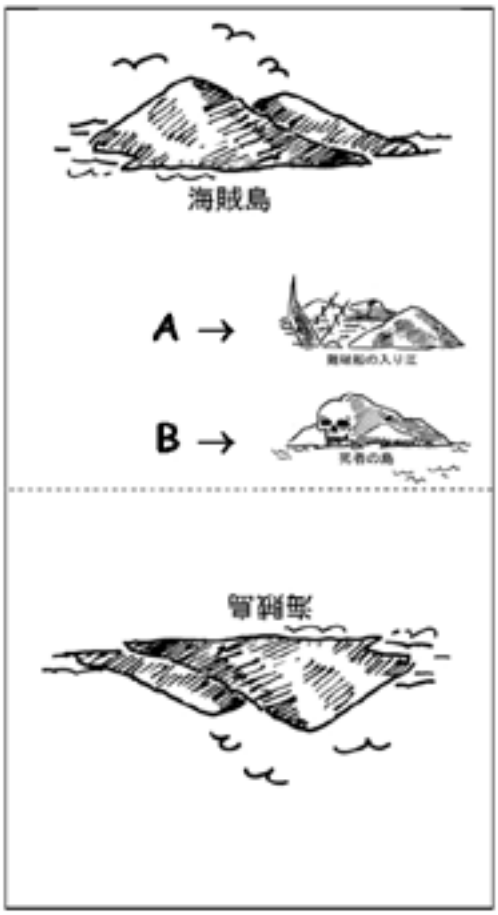
（注意：これは本番の学習で使うものとは違う地図です）

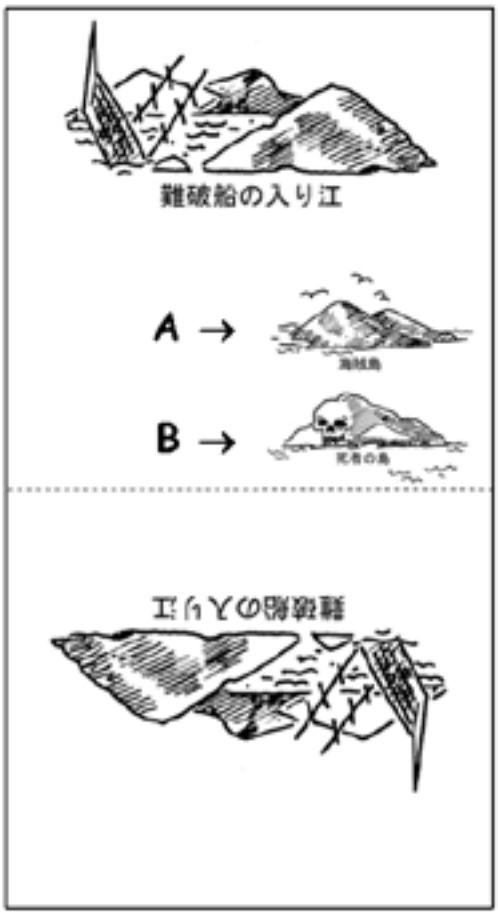
次の２ページにある３枚のカードを複写して、１枚ずつ違うカードを持たせます。
この３枚のカードは本番で使うものとは違うことを伝えます。

教師は海賊島からスタートして、船 A を選びます。海賊島のカードを持った子どもは船 A の行先である難破船の入り江を告げます。教師は地図にその経路を書き込みます。難破船の入り江で再び船 A を選びます。子どもは海賊島に戻るよう告げます。地図にそれを記します。今度は船 B を選びます。この経路を選ぶと死者の島へ行き、そこで動けなくなってしまいます。最終的な地図は、このようになります。



●実習カード





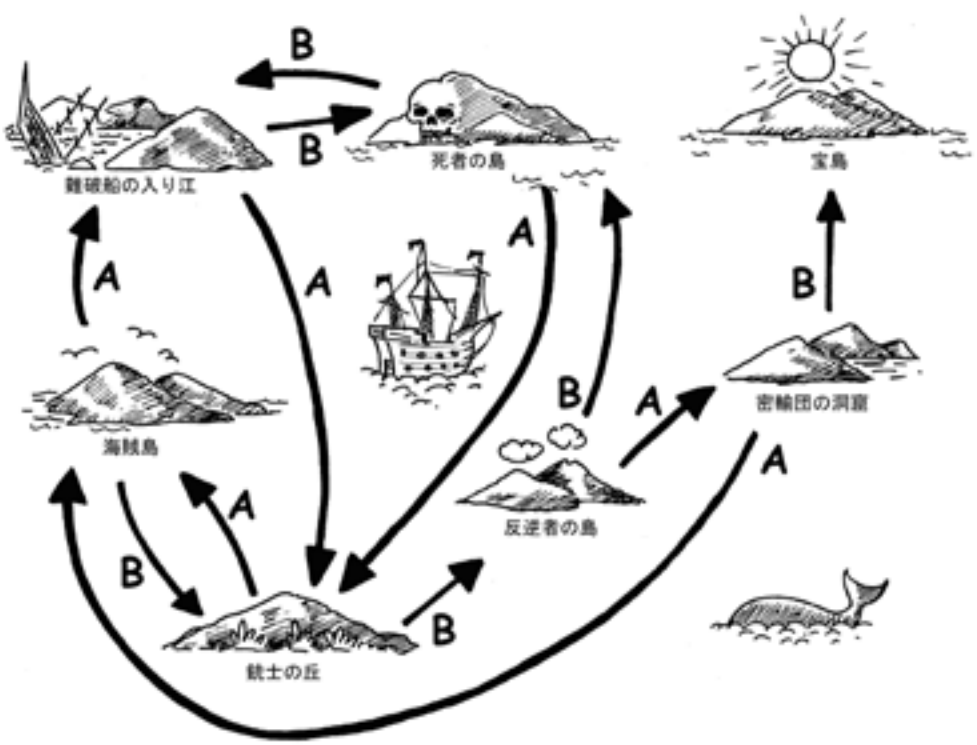
●実習カード

学習

7 人の子どもを選び、「島の住人」の役を与えます。子どもはそれぞれ自分の島のカードを持ちます。カードの裏面には、行先を示した秘密の命令が書いてあります。7 人は教室またはグラウンドで適当に散らばった位置に立ちます。残りの子どもにはまっさらな地図を配り、海賊島から宝島までの航路を注意深く書き込みながら航海するように指示します。（島に行く子どもは一度に 1 人だけに限定して、先に行先を耳にしないようにするのがよい考えでしょう。）

早く終わった子どもには、別の航路を探してみるよう指示します。

すべての航路を記した完全な地図は、次のようになります。



ふり返りの議論

最も早い経路はどれですか？ とても遅い経路はどれでしょう？

ループを含む経路もあります。その例を示すことができますか？

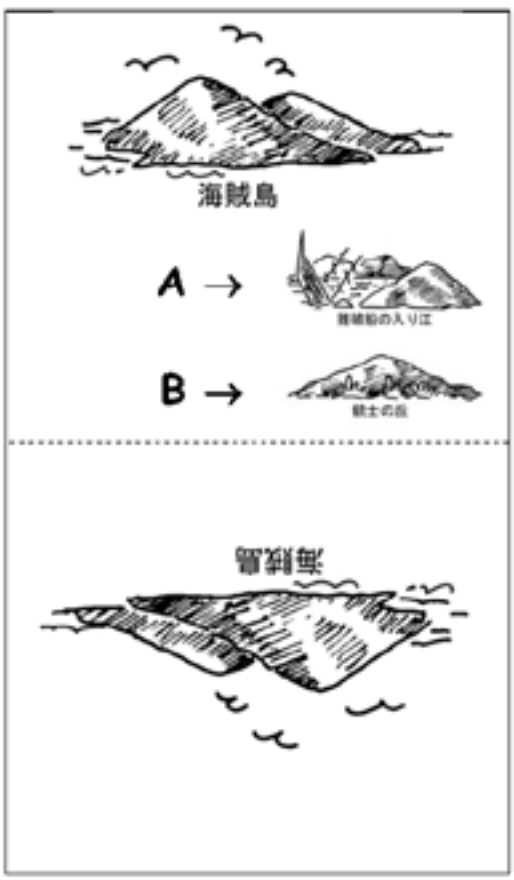
経路を記号で表してみよう。（たとえば、BBBABAB と BBBABBABAB はどちらも宝島に到着します）

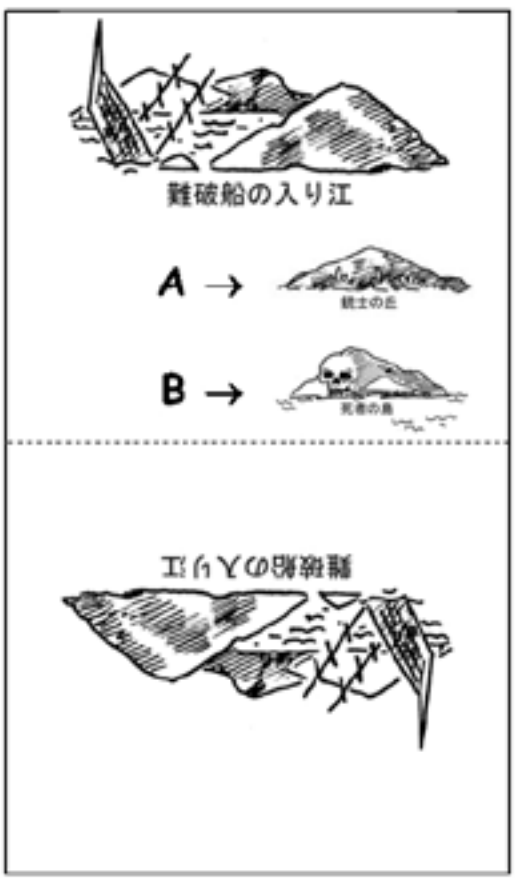
ワークシート

宝島の財宝までの行き方を見つけよう

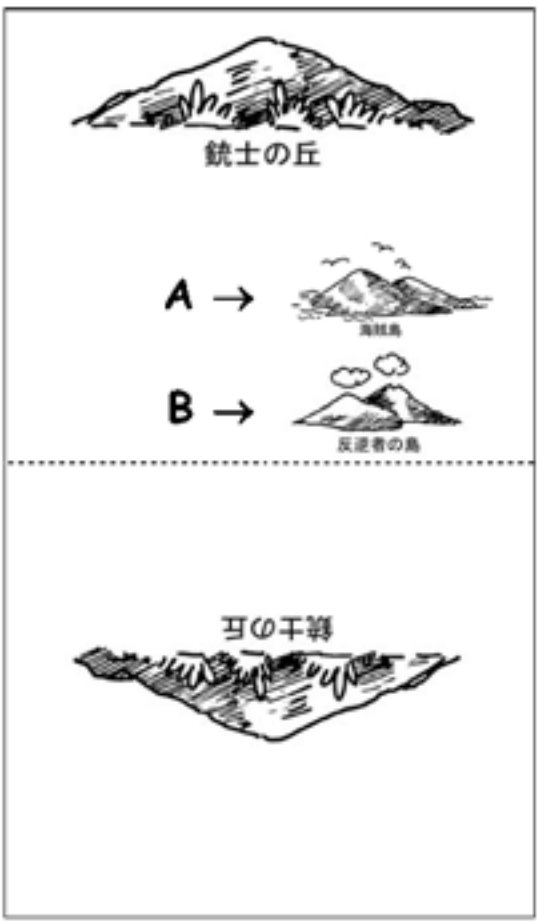
スタート（海賊島）、ゴール（宝島）

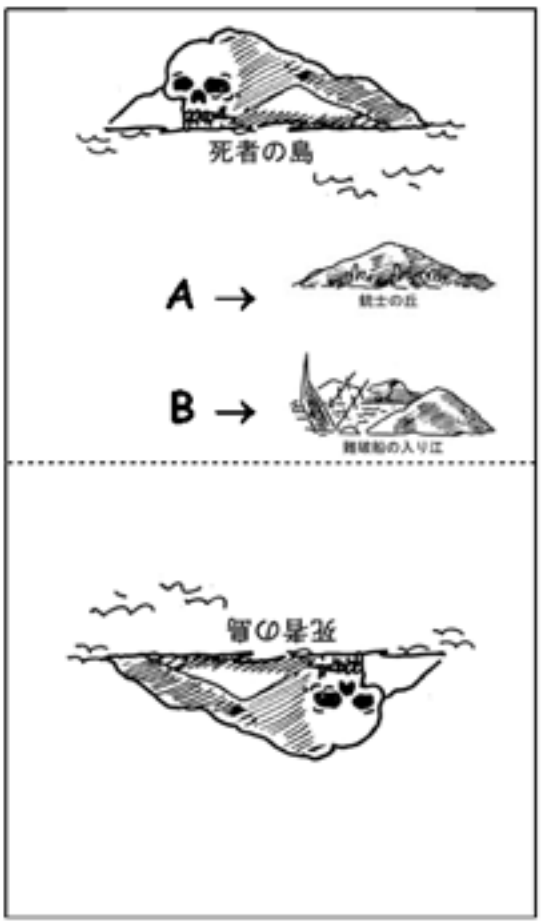
島のカード（1／4）



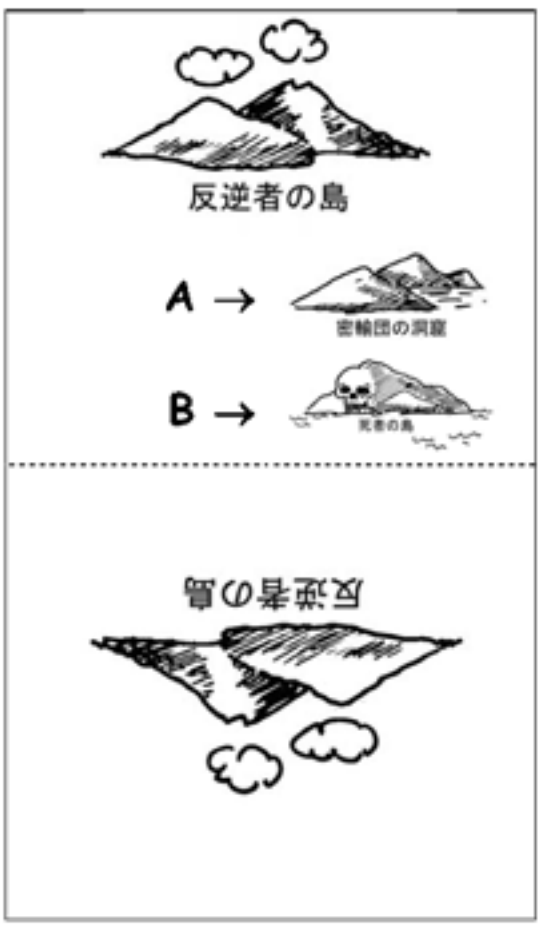


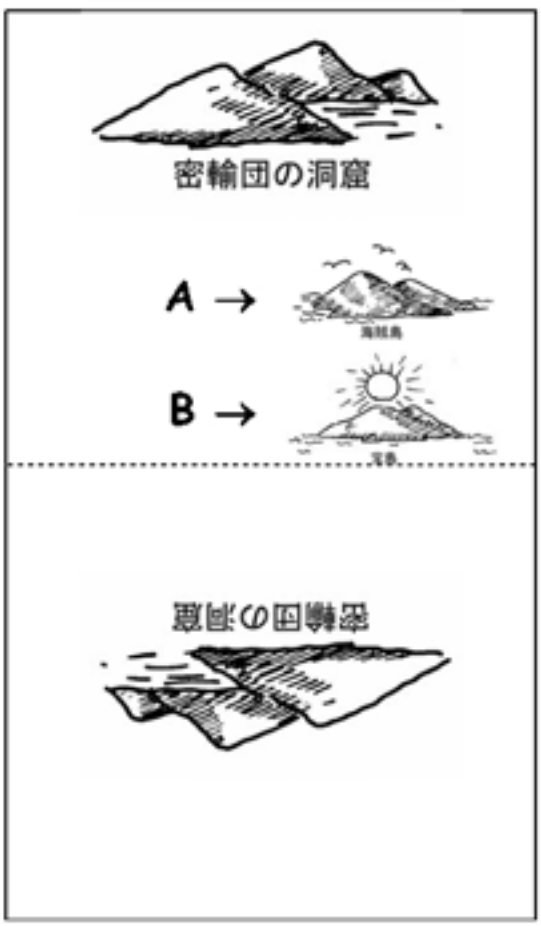
島のカード（２／４）





島のカード（3／4）

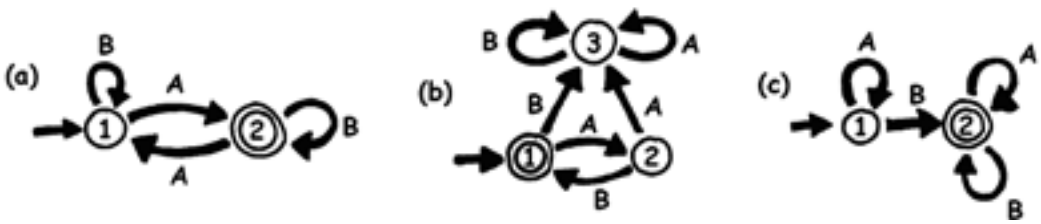




島のカード（４／４）

有限状態オートマトン

別の方法で地図を描くと、次のようになります。



島は番号が付された円で示され、目的の島（宝がある）は 2 重円で示されています。目的の島にたどり着くまでに、どのような経路をたどることができるでしょうか？

注意　地図（a）は、一連の移動に奇数回の A が含まれている場合にのみ 2 重円（2 番の島）で
終了します（例えば、AB、BABAA、AAABABA）。

地図（b）は、A と B の移動を交互に行うときのみ 2 重円に到達します（AB、ABAB、ABABAB、…）。

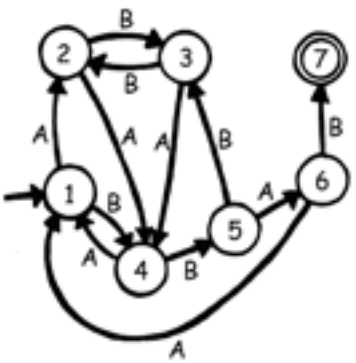
地図（c）は、一連の移動に少なくとも 1 つの B が含まれている必要があります（このよ
うなものは到達できません。A、AA、AAA、AAAA、…）。

ワークシート

宝島

埋めてあった宝物はうまく見つけられましたか？ 宝物を見つけるのはどのぐらい難しかったでしょう？ さあ、今度は自分だけの地図を作る時間です。

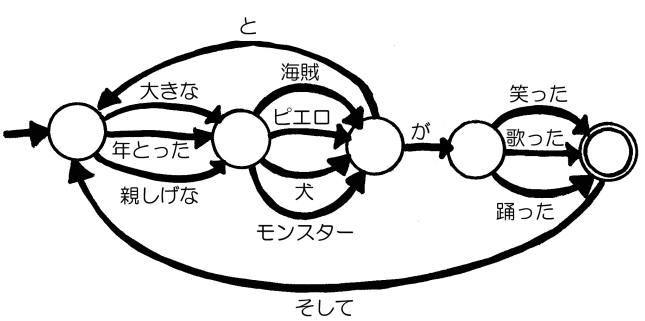
① これは、さきほどと同じ考え方で描いた複雑な地図です。計算機科学者は、パターンを表すために、このような手早く描ける簡単な経路の設計法を使います。



これを参考にして、オリジナルのゲームを作りましょう。海賊船がどんな経路を通るのかをはっきりさせて、自分だけの地図や島のカードを作ることができます。作った新しい地図で、宝島へ到着する最も短い経路はどれですか？

② 友達は、あなたの地図をうまくたどることができましたか？ 友達に A と B の文字の組み合わせを伝えて、正しい島へ到着できるかどうか確かめてみましょう。
この有限状態オートマトンの考え方をもとにして、さまざまなゲームやパズルを作り出すことができます。

③ 次に示すのは、地図上の経路を適当にたどり、出会った単語を記していくことで作文をする方法です。



さあ、この考え方を使って新しい作文ゲームを作りましょう。もしかしたら、笑い話だって作ることができるかもしれませんよ！

ワークシート

謎のコインゲーム

ある友達がインターネットからゲームをダウンロードしてきました。そのゲームは、ロボットがコインを投げて、それが表か裏かを当てるというものです。はじめ、そのゲームはとても簡単だと思われました。

「勝つ可能性は半々だから」みんなはそう考えました。ところが、実際にゲームをやってみると、どうも様子がおかしいことがわかってきました。どうやら、コインの表裏の出方には、規則性があるようです。ゲームに人間が勝てないような細工がされていた？ そんなはずはない！ そこで、調べてみることにしました。ジョーは次の結果が予想できるように、コインの表裏を書き出していきました。以下がその記録です：（h は表、t は裏）

hhthhthhhttthhhhttthttttthhhhhthhhttttthhhttt
hhhhhhttthttttttthttthttthhhttthhhthhhhhhhhh
ttthhhttttthhhhhttttttt

どんな規則があるか、わかりましたか？

とても単純なコイン投げの規則を表す「地図」があります。描けるかどうか、確かめてみましょう。（ヒント：たった４つの「島」しかありません）

実際のコンピュータでは

コンピュータでは、文字や入力の並びを処理するために有限状態オートマトンを使っています。

たとえば、ある電話サービスにダイヤルすると、「…には１を、…には２を、オペレータと直接お話になるには３を押してください」というメッセージが流れることがあります。このとき、電話の数字ボタンを押すことは、電話の向こうにあるコンピュータの有限状態オートマトンに入力を与えていることになります。もしシステムの設計ミスで有限状態オートマトンにおかしな繰り返しがあると、ぐるぐると同じ対話を繰り返し、電話した人間は、とんでもなくいらいらさせられるかもしれません！

次に、銀行の ATM からお金を引き出す場面を考えてみましょう。ATM に入っているコンピュータのプログラムは、一連の入力を処理して利用者に機能を提供するように作られています。可能性のあるすべての手順がプログラムの中に有限状態オートマトンとして表されています。利用者がキーを押すごとにオートマトンが次の状態に移ります。状態には、コンピュータへの指示を持つものもあります。たとえば「現金１万円を支払いなさい」「利用明細を印刷しなさい」「キャッシュカードを排出しなさい」などです。

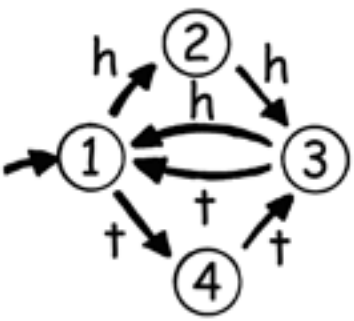
98 ページのような地図を使って文章を生成したり、利用者の入力を処理するプログラムもあります。1960 年代に、ある計算機科学者が人間と会話する有名な「イライザ（Eliza）」と呼ばれるプログラムを開発しました。この名称は、映画「マイ・フェア・レディ」でオードリー・ヘップバーンが演じた花売り娘の名前であるイライザ・ドリトル（Eliza Dolittle）に由来しています。プログラムは心理療法家のふりをして、「あなたのご家族について教えてください」「どうぞお続けください」といった質問をします。もっともらしい発話が行われるため、プログラムは実際には何も「理解」していないのに、本当に人間の心理療法家と対話していると考えてしまう人もいたそうです。

コンピュータは自然言語を取り扱うのは苦手ですが、人工言語を処理するのは得意です。プログラミング言語は重要な人工言語のひとつです。コンピュータは人間が書いたプログラムを読み込み、有限状態オートマトンを使ってコンピュータに適した命令の形式に変換します。

解答とヒント

謎のコインゲーム（99 ページ）

謎のコインゲームでは、コインを投げるときに次の地図を使っています。



この地図をたどれば、3 回ごとのコイン投げのうち、はじめの 2 回では必ず同じ面が現れることがわかるでしょう。

学習 12　出発進行

プログラミング言語

概要

コンピュータは普通、「言葉」でプログラムされています。その言葉は、コンピュータが理解できる言語で書かれています。プログラミングでイライラすることの 1 つは、意味のない結果が出る場合でも、コンピュータは常に言葉の指示に従うことです。この章では、プログラミングのこんな場面を子どもに経験してもらいます。

教科学習との関連

English: Interpersonal Listening Level 3

※国語　友だちのことばを聞く

技能

・Giving and following instructions.　命令を与える、命令を理解する。

年齢

・７歳以上

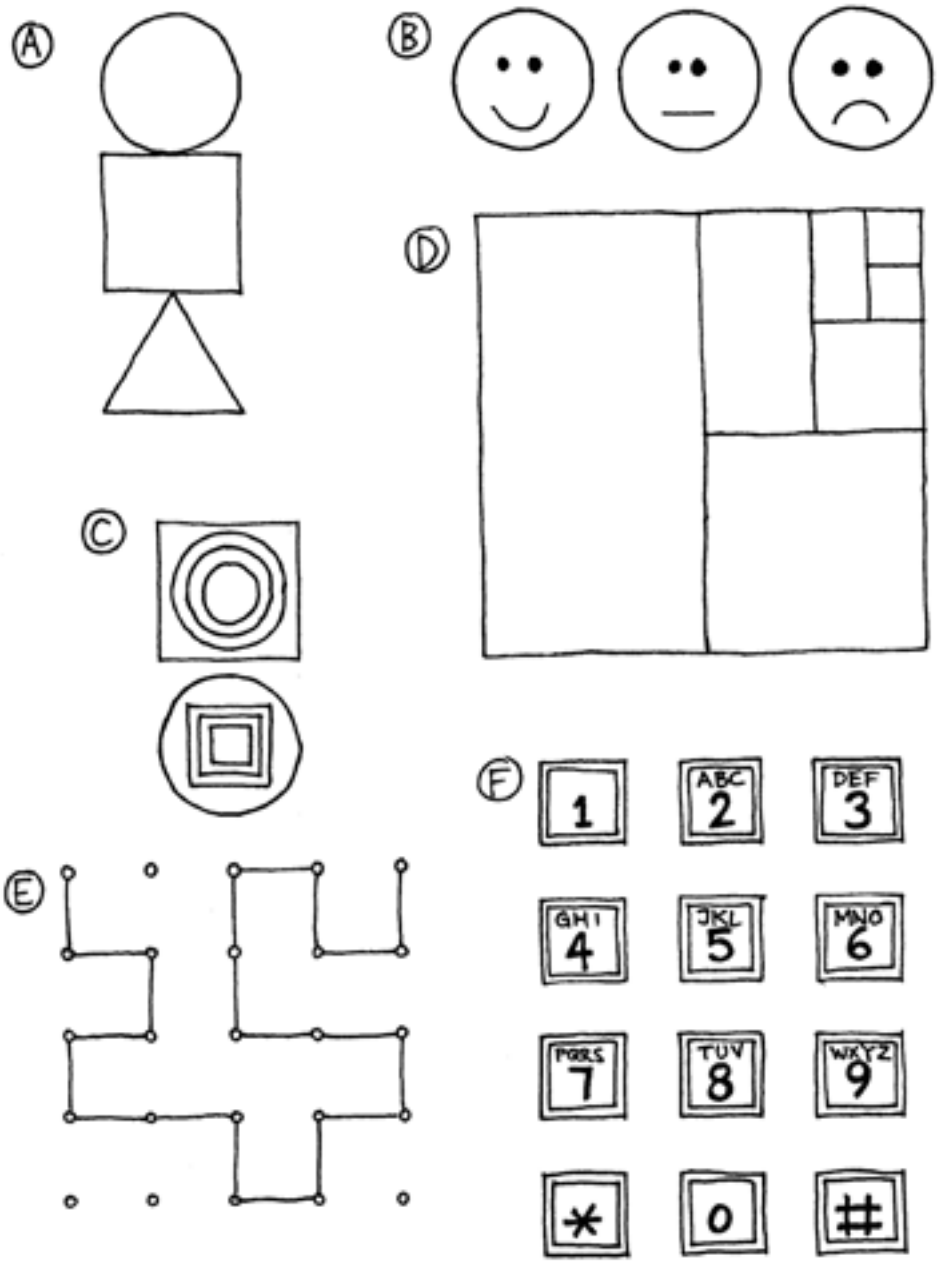
教材

＜先生に必要なもの＞

・次のページに示したような、絵が描かれたカード

＜子どもごとに必要なもの＞

・紙と鉛筆、定規


A
B
C
D
E
F
1
ABC 2
DEF 3
GHI 4
JKL 5
MNO 6
PQRS 7
TUV 8
WXYZ 9
*
0
#

出発進行

はじめに

人が指示通りに動くことが良いことなのかを話し合いましょう。 たとえば、もし君が閉まっているドアを指さされて「あのドアを通り抜けなさい」と言われたらどうする？

コンピュータは与えられた命令の通りに動きます。たとえ意味がないことでも！

実演の例

これらの指示で子どもが絵を描けるかを見てみましょう。

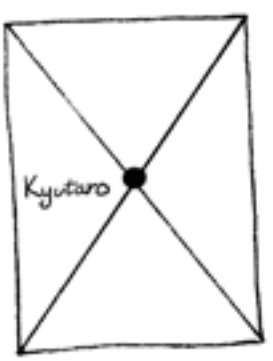
①　ページの真ん中に点を描きなさい。

②　ページの左上端から中央の点を通って、右下端に向かって直線を引きなさい。

③　ページの左下端から中央の点を通って、右上端に向かって直線を引きなさい。

④　ページの左側にできた三角の中に自分の名前を書きなさい。

その結果は、こんな感じになります。



活動

子どもを 1 人選び、(103 ページのような）絵を与えます。その子どもは、クラスの子どもが描くために絵を説明します。子どもは、指示を確かめるための質問をしてもかまいません。どれくらい早く正確に伝えられるかが目的です。

この練習を繰り返しますが、今度は、子どもは質問をしてはいけません。子どもは多くのことを長く覚えていられないので、簡単なイメージを使うのがよいでしょう。

次に、スクリーンの後ろに指示を出す子どもを隠してやってみましょう。質問はさせません。それで、指示（命令）だけが相手に伝わります。

この形のコミュニケーションは、プログラマがプログラムを書くときと似ていることを伝えましょう。彼らは、コンピュータに一連の指示を送りますが、実行してみるまでその結果はわかりません。

子どもに絵を描かせて自分たちの指示を書かせましょう。2 人組かクラス全体でやってみましょう。

バリエーション

① 紙飛行機を作るための指示を書きましょう。

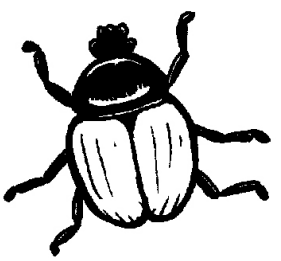
② 「1 メートル 歩く」「90 度 左回り」「90 度 右回り」のような指示を使って学校の中の秘密の場所にたどり着く方法を書きましょう。
子どもは、自分たちが考えた結果になるまで、指令が正しいかをテストして言葉の意味を考える必要があります。

③ 目隠しゲーム。目隠しした子どもに、教室の中で進む方向を指示してみましょう。

実際のコンピュータでは

コンピュータは行うべき指示（命令）の並びであるプログラムに従って動作しています。プログラムは特別に設計された言語で書かれており、コンピュータが理解できる命令が含まれています。用途によって適した言語は異なります。

どんな言語を使う場合でも、プログラマはコンピュータに実行させたいことを正確に伝える必要があります。人間とは違って、コンピュータはたとえ明らかに意味のない内容でも指示された通りに実行します。



プログラムがきちんと書かれていることは重要です。小さなエラーが、大きな問題を引き起こすことがあるからです。スペースシャトルの打ち上げや原子力発電所、鉄道の信号機などでコンピュータプログラムのエラーが起こす結果を想像してみてください。エラーは、「虫」と呼ばれますが、これは 1940 年代の電気計算機が故障したときに、中に入り込んでリレーを動かなくしていた 1 匹の蛾を取り除いたことに由来します。それ以来、エラーを除くことを「デバッグ」「虫取り」と呼ぶようになりました。

プログラムが複雑になるにつれて、エラーも多くなります。このことは、アメリカが戦略防衛構想（スターウォーズ計画）を進めているときに大きな問題となりました。これは核攻撃に対して防御する、コンピュータで制御されたシステムです。一部のコンピュータ科学者は、ソフトウェアに要求されている複雑さと信頼性が疑わしいことを理由にその計画は実現不可能であると主張しました。ソフトウェアはバグを見つけるためにテストをする必要がありますが、このシステムをテストすることは現実的ではありません。スターウォーズ計画のプログラムをテストするためには、アメリカ合衆国にミサイルを撃ち込む必要があったからです。

APPENDIX

Computer Science Unplugged 日本語版

追補

学習１　点を数える（２進数）

CD、CD-ROM、DVD と CD-R や DVD-R では記録の原理が少し違います。CD、CD-ROM、DVD では、情報の記録面に（レーザー光が入って来る面の反対側）に多数のピットと呼ばれる「出っぱり」があり、ピットがあるとそこで反射されるレーザー光と周囲の面で反射されるレーザー光が打ち消し合うことで反射光が弱くなることを利用してピットの有無を読み取ります。CD-R や DVD-R などでは記録時に色素の発色を変化させることで情報を書き込み、反射光はこの発色の違いを読み取ります。

コンピュータが２進数で文字の情報を表す仕組みは、「秘密のメッセージを送ろう」の課題と同様のものです。英語の文字を表すときは、本文にあるように ASCII コードを使います。ASCII コードには、英字（大文字、小文字）、数字、および括弧やピリオドやカンマなどの記号が含まれています。ASCII コードでは 7 ビットで 1 文字を表しますが、多くのコンピュータは 8 ビット（1 バイト）単位で情報を扱うのが便利なようにできているので、ASCII コードの 1 文字を 1 バイトに入れるのが普通です。

日本語では漢字を含めると多数の文字が必要なので、16 ビットのコードを使うのが普通です。漢字を扱える日本語の文字コードとしては、歴史的な事情などから、iso-2022-jp、euc-jp、Shift JIS など複数のものが使われています。このほか、UNICODE という世界各国の文字を一括して扱えるようにした文字コードも多く使われています。

学習２　色を数で表す（画像表現）

ファクシミリでピクセルの情報を圧縮するのに使われるのは、連続長符号化（Run Length Ecoding、RLE）という方法です。その原理は、たとえば 0 が 20 個続くのなら 0 を 20 個書く代わりに「(20,0)」のように繰り返しの数と値の組で表すことで短く表現するというものです。この方法は簡単で扱いやすいのですが、あまり圧縮率が高くありません。

写真などの場合、JPEG という画像ファイル形式（ディジタルカメラなどで多く使われます）では、「損失のある圧縮」という方法が使われます。損失のある圧縮とは、圧縮して伸長したときに完全には元の情報が復元できない方式で、その代わり圧縮率を高くできます。さらにこのとき、「損失が多くてもよいから圧縮率を高める」、「圧縮率はそれほど高くなくても損失を少なくする」などの選択が行えます。

画像ファイルを格納する形式でも、GIF や PNG と呼ばれる形式では損失のない圧縮が使われています。ただし、GIF の場合は使える色の数が最大 256 色という制限があるので、画像を GIF に変換すると色の微妙な違いが損なわれてしまうことがあります。

学習３　それ、さっきも言った！（テキスト圧縮）

データを圧縮する方式としては、画像や音などで使われる損失のある圧縮と、損失のない圧縮とがあります。画像や音で損失のある圧縮が使われるのは、元に戻したときに完全に元通りでなくても、人間が見たり聞いたりしたときに、だいたい元通りならそれで構わないからです。

しかし、コンピュータが扱うデータの多くのものでは、1 ビットでも変わってしまってはまずいので、損失のない圧縮が使われます。本文で言及している「たくさん使われる文字のコードを短くする方式」（ハフマン符号）、既に説明した連続長符号化（RLE）、ワークシートで扱っている LZ 符号のほかに、日本で考案された LHA のアルゴリズムなどもあり、絶えずより効果的に圧縮できる方式が研究され続けています。

学習４　カード交換の手品（エラー検出とエラー訂正）

本文では「ISBN 番号は 9 桁 +1 桁のチェックサムで一般の商品のバーコードとは違う」と説明しましたが、2004 年に ISBN 加盟国総会で次のような変更が決定され、2007 年から適用が開始されました。

・ISBN の桁数を 13 桁とし、一般の商品のバーコード（JAN コード）と一致させる。
・従来の 10 桁の番号は、先頭に「978」をつけて 13 桁にする。
・チェックサムの計算方法を変更する。

新しいチェックサムの計算方法は、次のようになります。

（1）（左から数えて奇数番目の桁の数字の合計）+（左から数えて偶数番目の桁の数字の合計 × 3）
　を求める

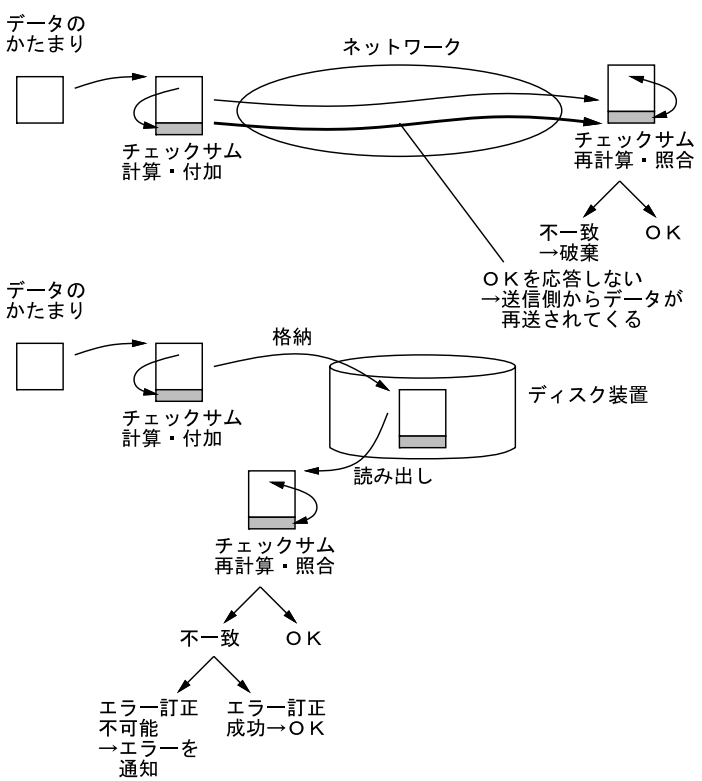


図１　ネットワークとディスク装置でのチェックサム

（2）合計の 1 桁目を 10 から引く

たとえば、10 桁の ISBN が「447413080X」の本は、13 桁の ISBN が「9784774130804」になります。最後の「4」がチェックサムで、その計算方法は次のようになります。

奇数桁 : 9 + 8 + 7 + 4 + 3 + 8 = 39
偶数桁 : (7 + 4 + 7 + 1 + 0 + 0) × 3 = 29 × 3 = 87
39 + 87 = 126 ←最後の 6 を取り出す
10 - 6 = 4　←これがチェックサム

現在出版されている本では、10 桁の ISBN と 13 桁の ISBN の両方が使われています。

現在、データが最も損なわれやすいのはネットワーク経由で長距離に渡ってデータを送る場合です。このような場合には、送出するデータのかたまりごとに、パリティを高度化した原理の循環冗長チェック（Cyclic Redundancy Check、CRC）という方法でチェックサム情報を付加し、受け取る側でこの情報をもとにチェックをすることでエラーを検出しています。ネットワークの場合、エラーが検出されたときは送り側に同じデータをもう 1 度送ってもらうことができますから、エラー訂正は行いません（図 1 上）。

一方、コンピュータのメモリ（主記憶）にデータを書き込むときや、ディスク装置にデータを書き込むときは、エラー検出だけでは不十分ですから（「もう 1 度送ってくれる」人がいませんから）、エラー訂正が行える方式を採用します（図 1 下）。ただし、メモリ素子でエラーが起きる頻度はそれほど大きくないので、安価なパソコンではメモリのエラー検出機能は省略されていることが多いかもしれません。

学習 5　２０の扉（情報理論）

コンピュータのソフトウェアを作るときには、「どういう場合には、どのように処理する」ということを洩れなく系統的に記述する必要があります（洩れがあると、たまたまその「考え落されていた」場合が起きたときに、とんでもないでたらめな処理をしてしまうかも知れませんから）。

このため、ソフトウェアの設計ときには、決定木や、それを表のような形で表した決定表（Decision Table）などを用いて、すべての場合が網羅されているかチェックしながら設計を進めることが行われます。また、決定木や決定表のデータをもとに、それに従った場合分けの処理を行うソフトウェアの構造を生成するツール（ソフトウェアを作ることを助けるソフトウェア）を使うこともあります。

学習 6　戦艦（探索アルゴリズム）

現在のコンピュータは極めて高速なので、処理の複雑さにもよりますが、データがメモリ（主記憶）に入っているなら、1 項目のデータを処理するのに必要な時間はたとえば 0.000001 秒（1 マイクロ秒）程度と考えてもよさそうです。だとすれば、遅い探索方式でもよいのでしょうか？データがディスクに格納されている場合、それを読み出してくるのに 0.01 秒くらい掛かるかも

しれないので、そのような場合は線形探索はやはり不向きだといえます。

データの量が増えたとき、処理にかかる時間がどのように増えるかを定式化したものを「時間計算量」といいます。たとえば、線形探索では、データの量に比例して処理に掛かる時間が増えますから、「線形時間」となります（図2の(a)）。

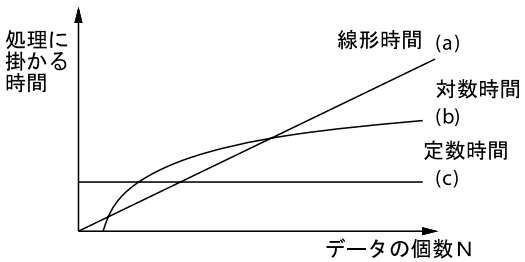


図2　探索アルゴリズムの時間計算量

2分探索では、データの量の対数に比例して処理に掛かる時間が増えますから、「対数時間」といいます。「対数」というのは、たとえば2の対数（2を底とする対数）であれば「2を何乗したらNになるか」の値のことをいいます。2の対数は、log 2 = 1、log 4 = 2、log 8 = 3 というふうに、Nが2倍になるごとに1ずつ増えて行きます（図2の（b））。線形時間であれば倍になるごとに時間も倍になるわけですから、それと比べるとずっと有利です。

ではハッシュ法ではどうでしょうか。ハッシュ法では、表を大きめにしておいて、衝突（複数のデータが同じ位置に入る）が起きる確率が十分に低いならば、ハッシュ関数を計算することで、一度で求めるデータが見つかります。したがって、表の大きさ（つまりデータの個数）によらず所要時間は一定です。つまり時間計算量は「定数時間」となります（図2の（c））。なにしろNの大きさに関係なく一定なのですから、対数時間よりも優ることになります。

学習7　いちばん軽いと一番重い（整列アルゴリズム）

整列（ソート）のアルゴリズムは、本文で説明しているようにさまざまなものがあります。これらも探索のときと同様、計算量を考えてみることができます。データの量がNの場合、挿入ソート、選択ソート、バブルソートでは平均してNの2乗に比例する処理時間が必要となります。なぜかというと、たとえば挿入ソートの場合、平均して現在の列の半分の長さだけ探していって挿入する場所を決めますから、（1 + 2 + …+ N）/2 ～ N ＊ N/4 でNの2乗に比例する手間となります。ですから、これらは遅いアルゴリズムです。

これに対し、クイックソートやマージソートは、N × log N（Nの2を底とする対数にNをかけた値）に比例する処理時間で済みます。たとえばクイックソートだと、列を分割する回数は（平均して半分くらいずつに分かれるとすると）log Nに比例します。それぞれの分割ごとにN個のデータを振り分けるわけですから、全体としてN × log Nに比例する手間となるわけです。これらは速いアルゴリズムです。

なぜこんな大まかなことで「遅い」とか「速い」とかいってしまえるのでしょうか？たとえば、10,000個のデータを整列するのに、挿入ソートで10ミリ秒、クイックソートで100ミリ秒の時間がかかったとします。クイックソートの方が10倍も遅そうですね。しかし、データの量がその千倍、つまり10,000,000個

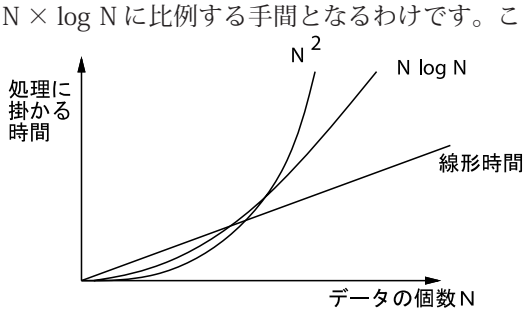


図3 整列アルゴリズムの時間計算量

になったとき、掛かる時間は挿入ソートだと 10,000 秒（1 ミリ秒の 1000 × 1000 倍）、クイックソートでは 1,000 秒（1 ミリ秒の 1000 × 10 倍… log 1024 = 10 なので）になると見積もられ、あっさり逆転してしまいます。この理由から、時間計算量が重要で、定数倍の違いはほとんど問題にならないことがわかります。この様子を図 3 に示します。なお、整列アルゴリズムの場合、線形時間で済ませることはできないことが知られています

学習 8　時間内に仕事を終えろ（並び替えネットワーク）

この章に出て来るような、整列のための節点（ノード）と辺の集まりのことを一般に整列ネットワーク（sorting network）といいます。どんなデータでも正しく整列が行えるようにするためには、きちんと設計された整列並べ替えネットワークが必要です（本文のネットワークを逆向きに使うとちゃんと整列できなかったことを思い出してください）。では、整列ネットワークは実際にどのようにして作られているのでしょう。

分かりやすい例として、バブルソートの整列ネットワークから見てみましょう。ここでは、比較のために逆順に並んだ 8 個の数を並べ替えることにします。

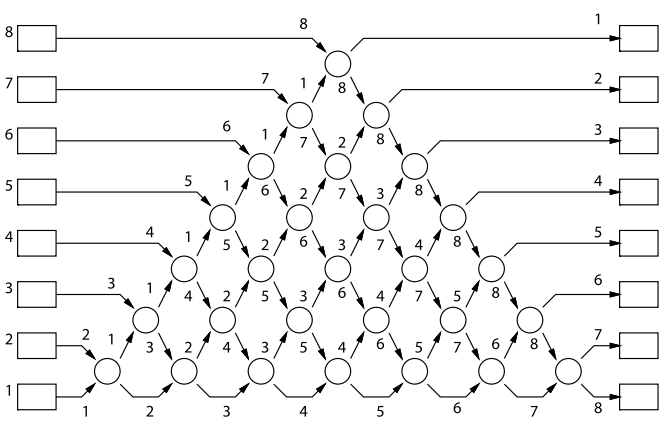


図 4 バブルソートの整列ネットワーク

バブルソートでは、隣同士の交換が全体として三角形のような形をしています。たとえば、いちばん小さな値である 1 は、隣の値と比較されるたびに三角形の左側の斜辺を登るように上に移動して行き、三角形の頂点まで達します。このように、軽い値（小さな値）は泡のように浮かび上がり、重い値（大きな値）は沈んで行くことから、この並べ替えの方法は「バブルソート」と呼ばれるようになりました。

バブルソートはわかりやすいのですが、たとえば 8 個の値を並べ替えるなら、7 + 6 + 5 + 4 + 3 + 2 + 1 = 28 個、N 個の値を並べ替えるなら N × （N − 1）÷ 2 個の比較交換場所が必要です（これは前の章でやった計算の手間と同じですね）。ただし「並列処理」ではこれらのうち互いに依存関係のないものは同時に行うことができるので、処理時間としては比較交換が N ＋（N − 3）回、8 個の値なら 13 段で行えます。つまり、並列処理によって「N の 2 乗の計算時間」を「線形時間」に改良できたわけです。

また、バブルソートで遊ぶことを考えると、全体が三角形の形をしているので、頂点に近いところで参加する子どもは、底辺に近いところで参加する子どもに比べて、少ししかゲームに

参加できない可能性があります。

そこで次に、本文で出て来た方法を紹介します。この方法の利点は、全員がほぼ均等に参加できることです。この方法では、まず 1 段目で隣どうしの比較を行い、次に（N-1）/2 段の「互い違いになった網目」の構造で最大値と最小値を求め、最後に（N-1）/2 段の三角形のネットワークで順に次の最大値と最小値を求めています。このため、最後の三角形のネットワークになる手前までは、どの子どもも同じようにゲームに参加できるわけです。

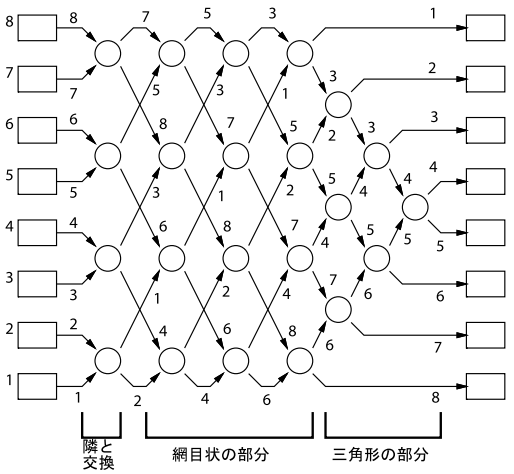


図 5 本文の整列ネットワーク

この方法では、8 個の値を並べ替えるのに 22 個の比較交換場所が必要で、これはバブルソートの 28 個より少なくて済みます。また、比較交換の段数も N-1 段で、8 個の値なら 7 段で行えますから、バブルソートの半分の時間で済みます。ただし時間計算量としては線形時間で、バブルソートと変わりません。

最後に、より効率のよい方法として、奇数偶数マージソート（odd-even merge sort）を紹介します。奇数偶数マージソートの考え方は、学習 6 で紹介されたクイックソートやマージソートと同様に、「分けて考える（分割して統治せよ）」に基づいています。8 個の値を並べ替えるときは、4 個ずつの値を並べ替えてから、それぞれの最小値と最大値を比較しています。たとえば、図のいちばん上のいちばん右の比較では、1 ～ 4 の最小値である 1 と 5 ～ 8 の最小値である 5 を比較することで、全体の最小値を決定しています。

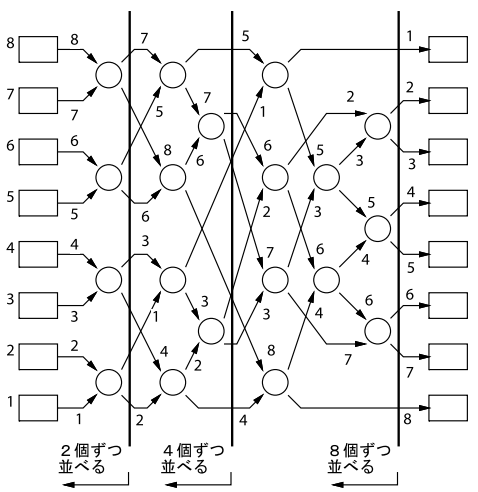


図 6 奇数偶数マージソートの整列ネットワーク

この方法では、離れた値同士の比較も行われます。バブルソートや本文で紹介された方法と違い、値は少しずつ移動するわけではありません。たとえば図の 1 の値は、たった 3 回の比較で最小と判定されています。ただし、移動が複雑になり、離れた場所との比較交換が必要になるので、子どもが地面に奇数偶数マージソートの図を描いたり、実際に動いて遊ぶのは難しいかもしれません。

この方法では、8 個の値を並べ替えるのに 19 個の比較交換場所で済みます。バブルソートの 28 個や先ほどの 22 個より少ないので、より効率の良い方法であることがわかります。

段数についても、8 個の場合は 6 段で先のものより少ないですし、一般に N 個の場合に $\log N \times (\log N - 1) \div 2$ と、先の 2 つよりも時間計算量が改良されています。

学習９　マッディ市プロジェクト（最小全域木）

最小全域木（Minimal Spanning Tree, MST）を求めるアルゴリズムは、本文でも説明されているようにいくつかの方法がありますが、ここではいちばんわかりやすい、プリム（Prim）という人が考えたアルゴリズムを説明します。以下ではグラフ理論のいい方に合わせて、接点のことを「頂点」、道のことを「辺」と呼びます。また道の横に書かれた数値は「重み」と呼びます。

プリムのアルゴリズムは、次のようになります。

(a) まず任意に 1 個の頂点を選ぶ。これが最初の部分木となる。

(b) 部分木に接している辺のうち、「(1) まだ部分木に含まれていない頂点に到達する、(2) 最も重みの小さい辺」を選び、木に加える。そのような辺が複数あれば、そのなかのどれか 1 つを任意に選んで加える。

(c) まだ木に含まれていない頂点があるなら、ステップ (b) に戻る。

プリムのアルゴリズムの中心的な考え方は、「現在の部分木の『どこからでも』新しい枝を伸ばしてよく、その中で重みが最小のものを選んで伸ばす」、というところにあります。実際に簡単なグラフでやってみましょう（図７）。この例では、たまたま頂点 A から始めるようにしてみました。

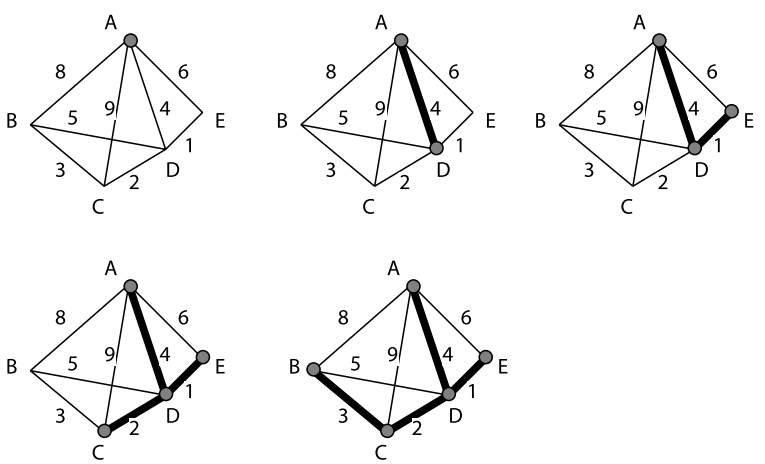


図7 プリムのアルゴリズムの実行例

学習１０　みかんゲーム（ネットワークにおけるルーティングとデッドロック）

デッドロックは並列性のあるシステムでは重要な課題となります。簡単な例を考えてみましょう。たとえば、工員の A さん、B さん、C さんが工具を使おうとしているとします。A さんは工具 X と Y が、B さんは工具 Y と Z が、C さんは工具 Z と X が同時に必要です。そこで一斉に工具を取りに行き、A さんは X、B さんは Y、C さんは Z を片手に持ちます。さて A さんは工具 X

は持ったので、工具Yを取ろうとしますが、それはBさんが使っているので待ちます。しかしBさんは工具Yは持っていますが工具Zも必要なので、Cさんが終わるのを待ちます。ところがCさんは工具Xも必要なのでAさんが終わるのを待ちます。3人とも待ってしまうので、仕事はこれ以上先に進みません（図8）。

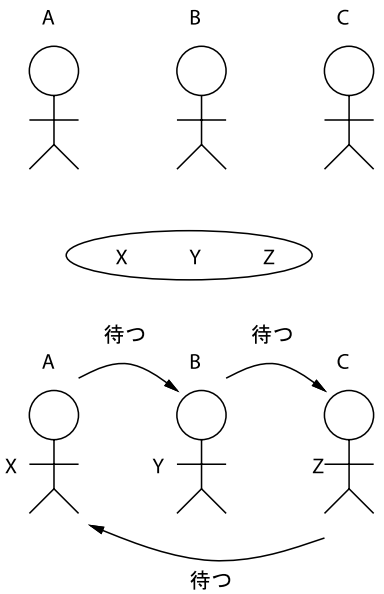


図8 デッドロックの例

このように、「AはBが終わるのを待ち、BはCが終わるのを待ち、CはAが終わるのを待つ」のような「堂々めぐり」の関係ができることをデッドロックと呼びます（3人だと「3すくみ」ともいいますね）。人数は3人に限らず、2人や4人、そして何人の場合でも起こり得ます。

デッドロックがおきると、関連する処理は止まったまま先に進まなくなります。人間がみかんゲームをする場合は「あれ、行き詰まったかな」と思って別のやり方に切替えるので止まったままにはなりませんが、普通のコンピュータのプログラムにはそのような「臨機応変の知恵」はありませんから、本当に止まったままになります。このため、あらかじめデッドロックを防ぐ方法を組み込んでおく必要があります。

デッドロックを防ぐ方法のひとつに、「必要なものをロックする順序をすべての人が同じ順序にする」というものがあります。たとえば、「工具は必ずX、Y、Zの順に取ること。たとえXが必要がなくてもYを取るならXを先に取り、Zしか必要がなくてもXとYも取ること」と決めておきます。こうしておけば、Xを取ったAさんがYを取ろうとしたら先に取られているということはあり得ませんから、堂々めぐりはなくなるわけです。

さて、「みかんゲーム」は、どのようにしたら効率よく目指す状態に到達できるでしょうか。それには、「状態」と「状態の間の移り変わり」に着目します。このゲームでは「状態」とは「誰がどのみかんを持っているか」を意味します。そして「移り変わり」は、「誰から誰にどのみかんを受け渡すか」に対応します。

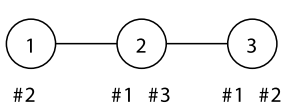


図9 みかんゲームの状態の例

たとえば、図9に3人が1列に並んだみかんゲームの1つの状態を示します。1番の人はみかん#2、2番の人はみかん#1と#3、3番の人はみかん#1と#2を持っています。

これをもっとコンパクトに表現するため、図10の左上隅の箱のように書くことにします。これが1つの「状態」です。この状態でできることは、みかん#1が2番から1番に渡されるか、またはみかん#3が2番から1番に渡されるかです。これを「1:2-1」「3:2-1」のように書くことにしました。その結果はそれぞれ違った状態になりますから、最初の状態からその2つの状態それぞれに矢線を引き、そこに渡し方を記入しておきます。渡し方はその2つしかありませんから、最初の状態から出る矢線はこの2つ以外にあり得ないことに注意してください。

「1:2-1」を選んだ場合の状態（中央上の状態）では、「1:1-2」「2:1-2」「1:3-2」「2:3-1」の4つの渡し方がありますから、矢線が4つ出ています（そのうち1つはさっきの状態に戻ってしまいます！もらったみかんをまた元の人に返せばさっきと同じになりますから…）このようにし

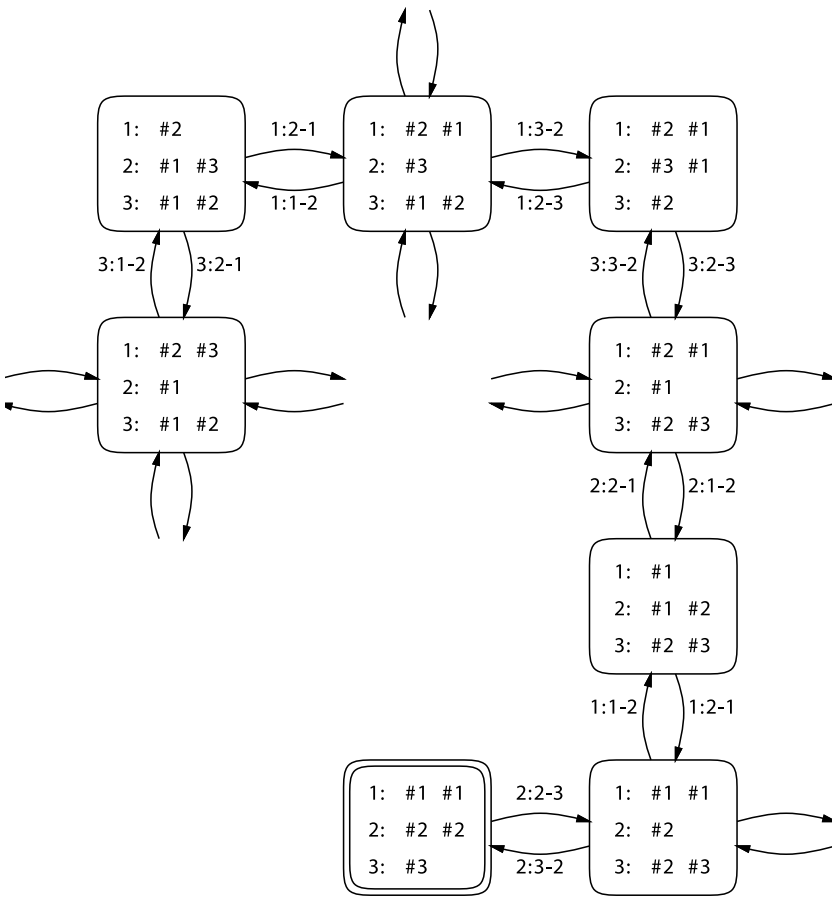


図 10 みかんゲームの状態遷移図の例

て、たどり得る限りの状態を尽くして行くことで、めざす状態（二重の四角で表してあります）に到達する経路が分かります。みかんゲームでは必ずめざす状態に到達できますが（その理由は考えてみてください）、問題の種類によってはめざす状態には決して到達できないとわかることもあります（これ以上新しい状態を増やすことは不可能なのに、状態のなかにめざす状態がない場合）。

このように、問題を状態と状態遷移の形に整理することで、問題の構造がわかりやすくなりますし、確実に目的の状態に到達する方法がはっきりするでしょう。この考え方は次の章（学習 11）でも出てきます。

みかんゲームの状態遷移図でポイントとなるのは、ある番号のみかんは（1 つを除いて）2 個ずつありますが、そのどちらであるかは問題を解く上では関係がないので状態として区別しないこと、またある番号のみかんを右手で持っているか左手で持っているかも同様に、問題を解く上では関係がないので状態として区別しないことです。これらを区別していると、状態の数がずっと多くなってごちゃごちゃになってしまうでしょう。つまり、問題を解く上で問題になるような違いがある場合だけを状態として区別することが大切です。

学習１１　宝探し（有限状態オートマトン）

有限状態オートマトン（Finite State Automata、FSA）はコンピュータ科学における強力で有力な定式化の方法の 1 つです。たとえば、「コンピュータに与える文字の並びがきちんと数値の形をしているか」を調べることを考えます。そんなの簡単ですか？ たとえば次のものでどうでしょう。

・数字（0 ～ 9）が 1 個以上並んだものが数値である。

しかし、負の数だったら数値の前にマイナス符号があるはずです。正の数でもお望みならプラス符号を書くかもしれません。では次のものではどうでしょう。

・プラス符号か、マイナス符号があってもなくてもよい。その後に、数字（0 ～ 9）が 1 個以上並んだものが、数値である。

しかし小数点のついた数もあるはずです。

・プラス符号か、マイナス符号があってもなくてもよい。その後に、数字（0 ～ 9）が 1 個以上並んだものが、数値である。途中に「.」があってもよい。

途中っていうのは、「-.5」みたいなのも含まれるでしょうか。それはあまりよくなさそうです。「0.1.2」のように 2 回現れるのも困ります。そうすると、次のように正確に書く必要があるでしょう。

・プラス符号か、マイナス符号があってもなくてもよい。その後に、数字（0 ～ 9）が 1 個以上並ぶ。それで終わってもよいが、「.」があって、さらにその後に数字が 1 個以上並ぶということでもよい。

ごちゃごちゃになりましたね。このようなものも、FSA を使うときれいに定式化できます。図 11 の FSA は、上で説明したような数式を認識します。これを左上の入口から文字の並びに対応してたどっていき、最後に◎のところで終われば、文字の並びは正しい形式の数値というこ

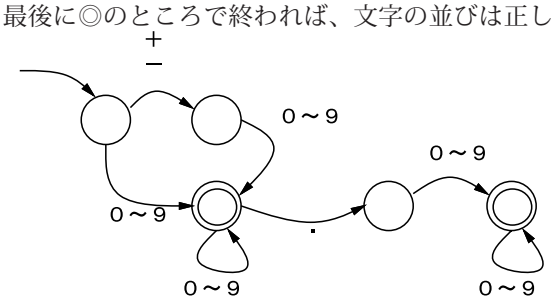
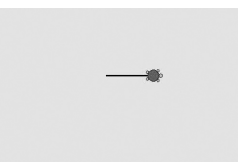


図１１ 数値を認識する FSA

とです。

学習１２　出発進行（プログラミング言語）

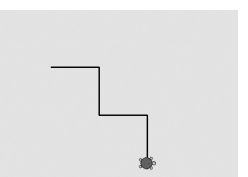
本文に書かれているように、コンピュータのプログラムはプログラミング言語という決まった形式で記述します。高度なシステム開発には、専門家が十分勉強した後で使うことが前提となるようなプログラミング言語が使われますが、コンピュータ教育の一環としてプログラミングを体験するのなら、もっと簡潔で分かりやすいプログラミング言語を使うことがおすすめです。



たとえば、ドリトル*というプログラミング言語をちょっとだけ見てみましょう。一番簡単なプログラムは次のような感じです。

```
カメ＝タートル！作る。
カメ！ 100　歩く。
```

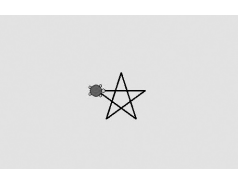
1 行目でカメという名前のもの（これはタートルという種別で、命令して歩かせるとその歩いた軌跡の線が画面に現れます）を用意します。2 行目でそのカメを歩かせます。これで「100 歩」ぶんの直線が画面に引かれます。



もう少し複雑な図形にしてみましょう。次のではどうでしょうか。

```
カメ＝タートル！作る。
カメ！ 100　歩く　90　右回り。
カメ！ 100　歩く　90　左回り。
カメ！ 100　歩く　90　右回り。
カメ！ 100　歩く　90　左回り。
```

曲がりながら歩くことで、ぎざぎざの線を画面に引くことができます。



こうやって動き方を逐一指示する代わりに、「くりかえし」を利用することで、短いプログラムでも一気にたくさんの線を引くことができます。

```
カメ＝タートル！作る。
「カメ！ 100　歩く　144　右回り」！ 5　繰り返す。
```

このようにして、コンピュータに命令することでさまざまな作業をさせることができます。

＊プログラミング言語「ドリトル」http://dolittle.eplang.jp/

索引

２進数…… 3,108
２進法…… 4,12
ASCII …… 11,108
ISBN …… 34,109
圧縮…… 21,23,30,41,108
アルゴリズム…… 44
アンプラグド…… Ⅲ
エラー…… 31,36,106,110
エントロピー…… 41
奇数偶数マージソート…… 113
クイックソート…… 67,69,111
グラフ…… 77,79,114
経路制御…… 83
決定木…… 40,110
コード（符号）…… 8,9
最小全域木…… 80,114
時間計算量…… 111,113
循環冗長チェック…… 110
状態遷移…… 116
情報…… 2
情報量…… 38,41
伸張…… 30
整列…… 64,69,111
整列ネットワーク…… 112
線形時間…… 111
線形探索…… 47,110
選択ソート…… 66,69,70,111
挿入ソート…… 68,69,111
ソート…… 64,111
対数時間…… 111
探索アルゴリズム…… 45,110
逐次処理…… 74
チェックサム…… 35,110
定数時間…… 111
データ…… 2,3,23,31
デッドロック…… 81,83,114
ドリトル…… Ⅲ ,118
並び替えネットワーク…… 71,75,112
二分探索…… 48,63
バーコード…… 34,63,109
バイト…… 12,108
ハッシュ法…… 45,49,63,111
ハフマン符号…… 109
バブルソート…… 68,111,112
パリティ…… 33,36,110
ピクセル…… 15,21,108
ビット…… 11,12,41,108
ファクシミリ…… 9,21
輻輳…… 83
プログラム…… Ⅲ ,86,106,118
並列処理…… 74,75,112
マージソート…… 68,111
モデム…… 9,21
有限状態オートマトン（FSA） 97,100,117
ルーティング…… 83,114
連続長符号化…… 108

【監訳者紹介】
兼宗　進
1963 年東京生まれ。87 年千葉大学工学部電子工学科卒業、89 年筑波大学大学院理工学研究科修士課程修了。
企業勤務ののち、2004 年筑波大学大学院ビジネス科学研究科博士課程修了、博士（システムズ・マネジメント）。
2004 年より一橋大学総合情報処理センター准教授。
主な研究対象分野はコンピューター教育。教育用プログラミング言語「ドリトル」の開発も手がける。

【翻訳者】
正田　良：国士舘大学文学部教授
鎌田敏之：愛知教育大学教育学部准教授
紅林秀治：静岡大学教育学部准教授

【翻訳協力者】
西田知博：大阪学院大学情報学部講師
井戸坂幸男：三重県松阪市立飯南中学校教諭
保福やよい：神奈川県立松陽高等学校教諭

【追補執筆者】
久野　靖：筑波大学大学院教授

コンピュータを使わない情報教育
アンプラグドコンピュータサイエンス

2007 年　9 月 1 日　　第 1 刷　発行
2008 年　8 月 1 日　　第 3 刷　発行

著　者　Tim Bell,Ian H.Witten and Mike Fellows
監訳者　兼宗　進
訳　者　正田　良、鎌田敏之、紅林秀治　他
発行人　原　久太郎
発行所　株式会社 イーテキスト研究所
〒 113-0033 東京都文京区本郷 5-1-16
Tel 03-3815-3037　Fax 03-3818-1219
http://www.etext.jp
ISBN978-4-904013-00-7

装丁／林　健造