アルゴリズムとデータ構造
第2週

掛下 哲郎
kake@is.saga-u.ac.jp

前回のポイント

- 前回のポイント
  - 「アルゴリズム」概念の理解
  - 「データ構造」概念の理解
  - アルゴリズムを評価するための尺度と、その評価方法
  - Big O 記法の定義と具体例

講義スケジュール

週	講義計画
1ー2	導入
3	探索問題
4ー5	基本的なデータ構造
6	動的探索問題とデータ構造
7	アルゴリズム演習（第1回）
8ー9	データの整列
10ー11	グラフアルゴリズム

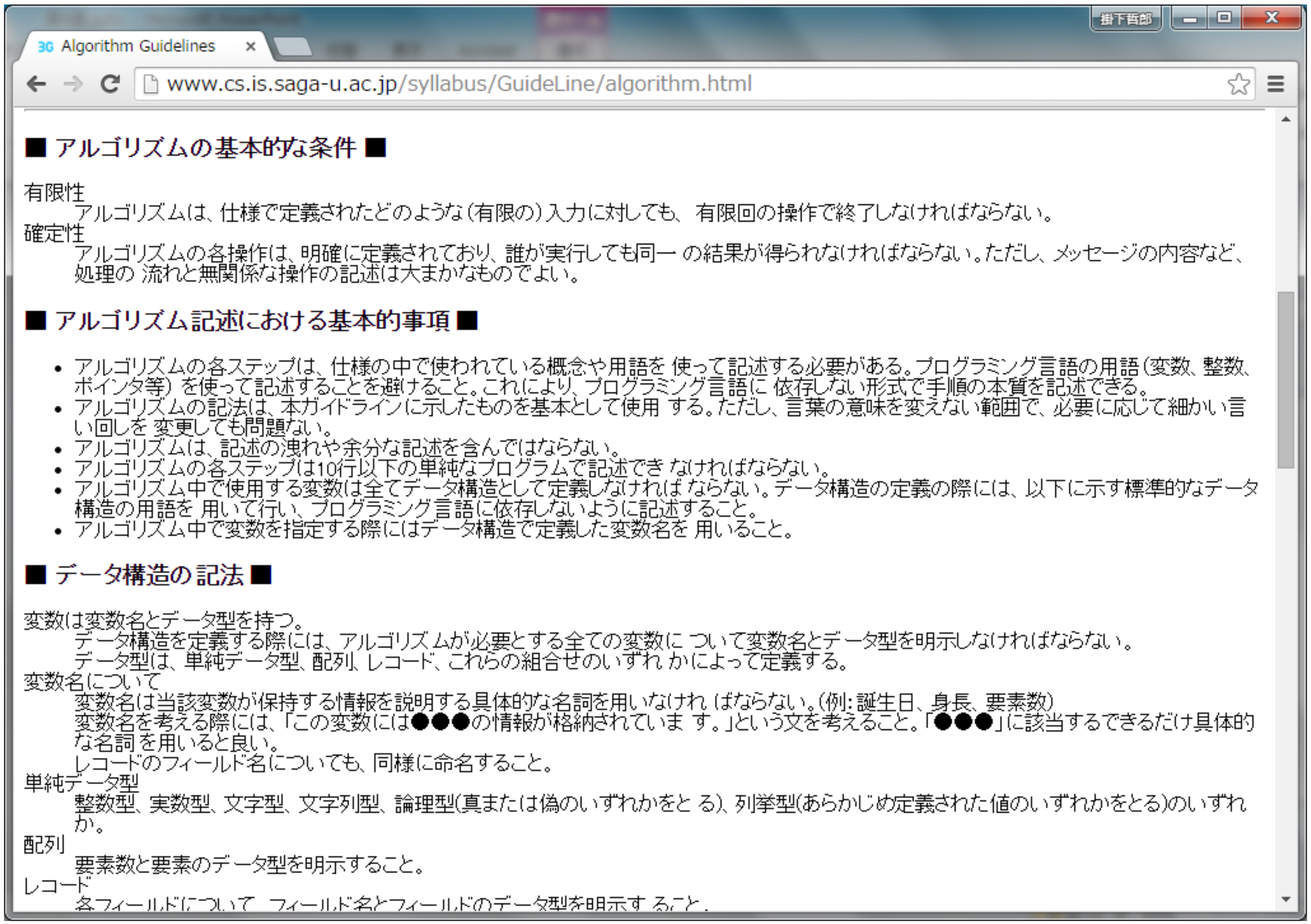
アルゴリズム作成のガイドライン

アルゴリズムの良し悪しを具体例で見る

- 具体的なアルゴリズムを取り上げ，改良してゆく

アルゴリズム作成のガイドライン

http://www.cs.is.saga-u.ac.jp/syllabus/GuideLine/algorithm.html



■ アルゴリズムの基本的な条件 ■

有限性

アルゴリズムは、仕様で定義されたどのような(有限の)入力に対しても、 有限回の操作で終了しなければならない。

確定性

アルゴリズムの各操作は、明確に定義されており、誰が実行しても同一 の結果が得られなければならない。ただし、メッセージの内容など、処理の 流れと無関係な操作の記述は大まかなものでよい。

■ アルゴリズム記述における基本的事項 ■

- アルゴリズムの各ステップは、仕様の中で使われている概念や用語を 使って記述する必要がある。プログラミング言語の用語(変数、整数、ポインタ等) を使って記述することを避けること。これにより、プログラミング言語に 依存しない形式で手順の本質を記述できる。
- アルゴリズムの記法は、本ガイドラインに示したものを基本として使用 する。ただし、言葉の意味を変えない範囲で、必要に応じて細かい言い回しを 変更しても問題ない。
- アルゴリズムは、記述の洩れや余分な記述を含んではならない。
- アルゴリズムの各ステップは10行以下の単純なプログラムで記述でき なければならない。
- アルゴリズム中で使用する変数は全てデータ構造として定義しなければ ならない。データ構造の定義の際には、以下に示す標準的なデータ構造の用語を 用いて行い、プログラミング言語に依存しないように記述すること。
- アルゴリズム中で変数を指定する際にはデータ構造で定義した変数名を 用いること。

■ データ構造の 記法 ■

変数は変数名とデータ型を持つ。

データ構造を定義する際には、アルゴリズムが必要とする全ての変数に ついて変数名とデータ型を明示しなければならない。

データ型は、単純データ型、配列、レコード、これらの組合せのいずれ かによって定義する。

変数名について

変数名は当該変数が保持する情報を説明する具体的な名詞を用いなけれ ばならない。(例: 誕生日、身長、要素数)

変数名を考える際には、「この変数には●●●の情報が格納されていま す。」という文を考えること。「●●●」に該当するできるだけ具体的な名詞 を用いると良い。

レコードのフィールド名についても、同様に命名すること。

単純データ型

整数型、実数型、文字型、文字列型、論理型(真または偽のいずれかをと る)、列挙型(あらかじめ定義された値のいずれかをとる)のいずれか。

配列

要素数と要素のデータ型を明示すること。

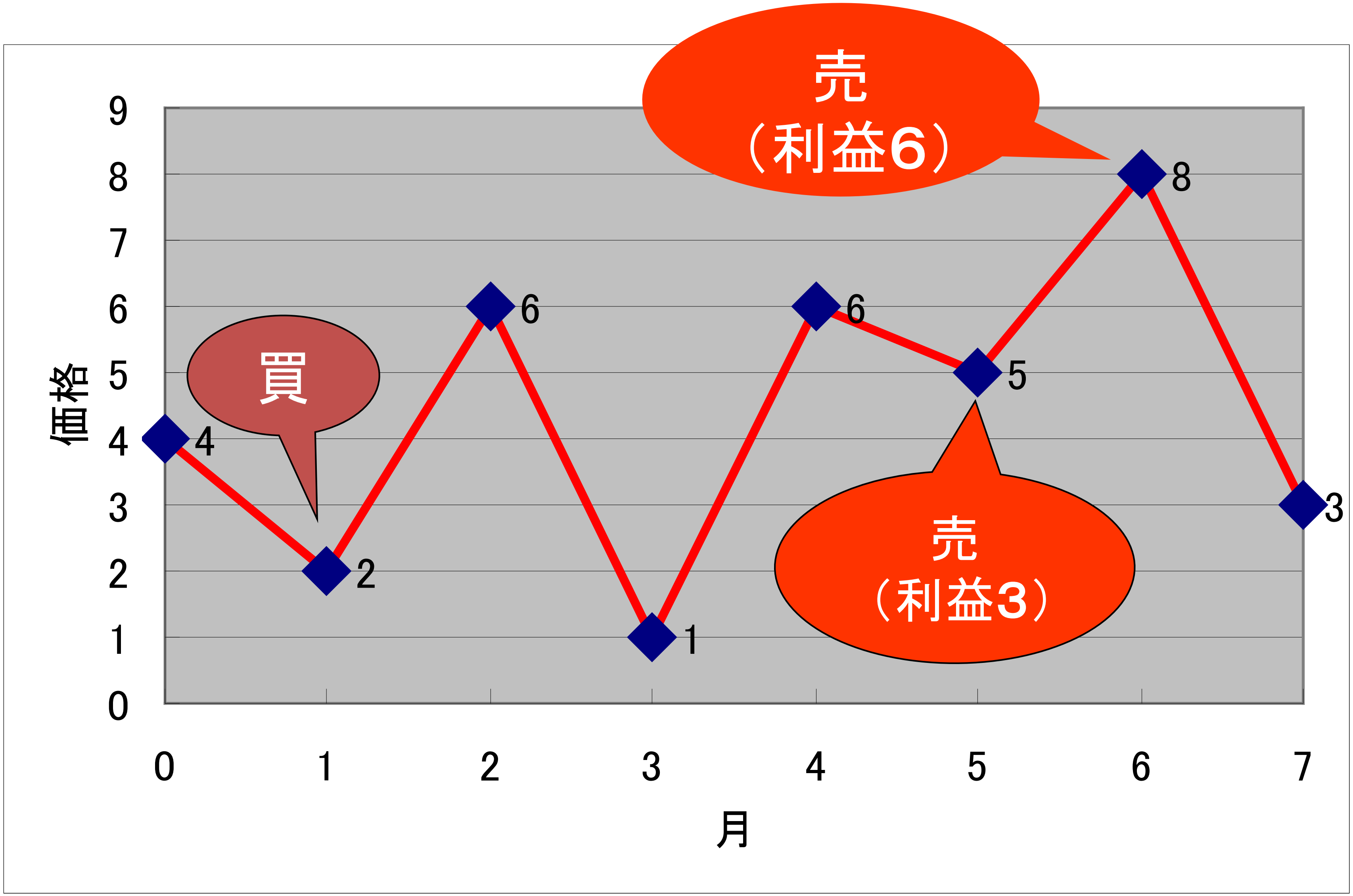
レコード

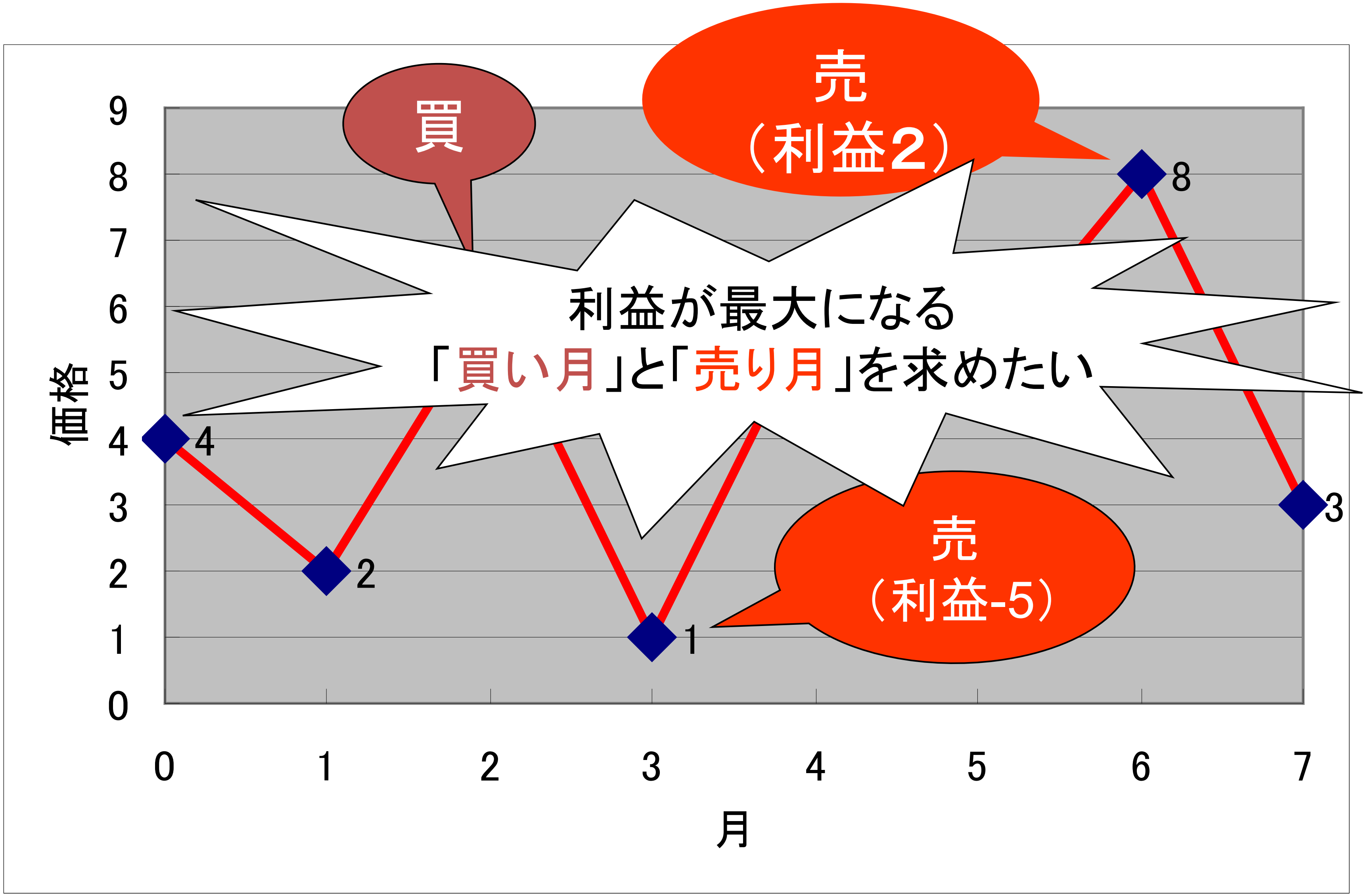
各フィールドについて、フィールド名とフィールドのデータ型を明示す ること。

株取引に関するアルゴリズム

- 状況:
  - あなたは証券会社のセールスマン
  - 顧客に「株の魅力」をアピール(「儲かります！」)
  - 株式投資の「最大売却益」を求めたい

    (利益) = (売値)–(買値)


売
（利益6）
買
売
（利益3）
価格
月
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
4
2
6
1
6
5
8
3


買
売
（利益2）
利益が最大になる
「買い月」と「売り月」を求めたい
売
（利益-5）
価格
月
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
4
2
1
8
3

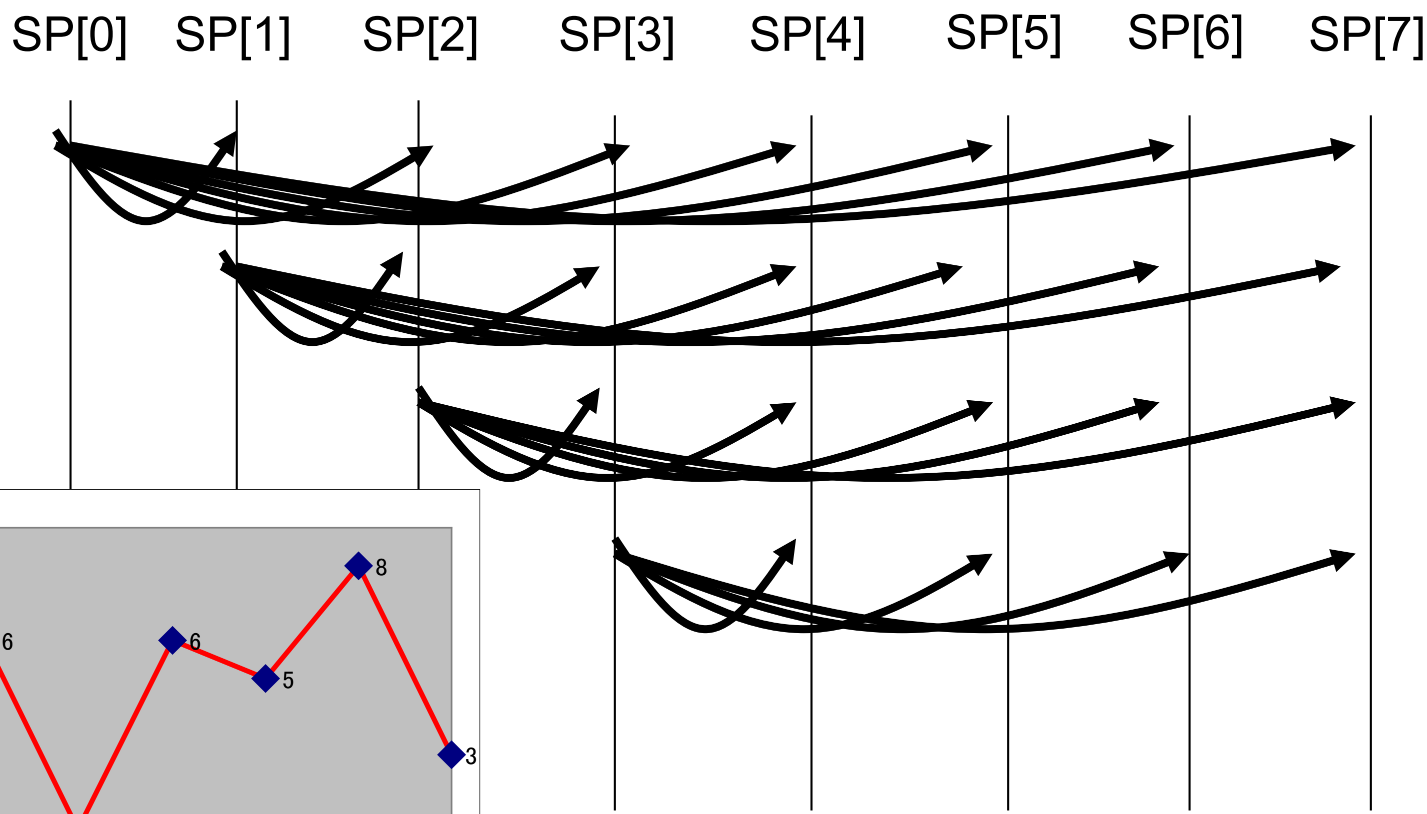
どうやって解くか？

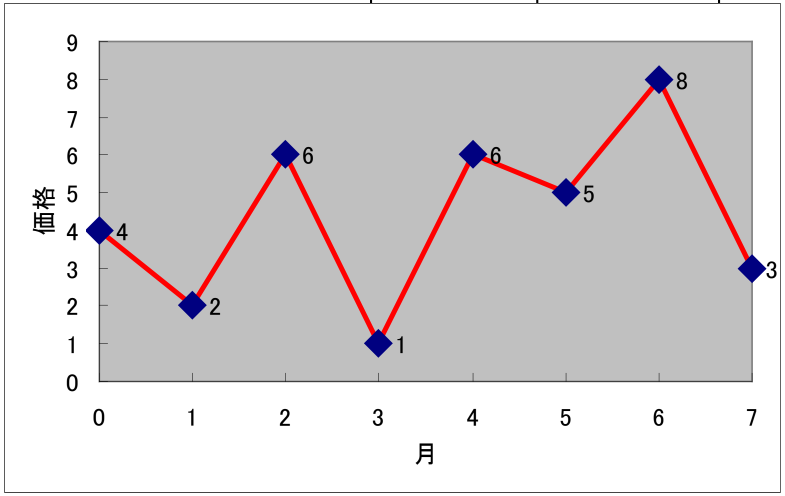
- 株価（Stock Price）を，配列に格納する
  SP[i]はi番目の月の株価を保持

配列SP	4	2	6	1	6	5	8	3

- 全ての i < j で，SP[j] – SP[i] を求め，最大値を探せば良い

アルゴリズム1.1(総当たり)





アルゴリズム1.1(総当たり)

1. 利益最大値 を 0 に初期化する.
2. 買月を 0 から n-2 まで変えながら以下の処理を繰り返す.
   - 2-1. 売月 を 買月+1 から n-1 まで変えながら以下の処理を繰り返す.
     - 2-1-1. 利益(売月の株価 –買月の株価)が利益最大値を上回っていれば, 利益最大値を利益で更新する.
3. 利益最大値を返す

トレース表：変数値の変更履歴　n=8

ステップ	買月	売月	備考
1			
2	0		
2-1	0	1	
2-1-1	0	1	
2-1	0	2	
2-1-1	0	2	
2-1	0	3	
2-1-1	0	3	
:	:	:	
2-1	0	7	7回繰り返し
2-1-1	0	7	

ステップ	買月	売月	備考
2	1		
2-1	1	2	
2-1-1	1	2	
	:	:	
2-1	1	7	6回繰り返し
2-1-1	1	7	
	:	:	
2	6		
2-1	6	7	1回繰り返し
2-1-1	6	7	
3			

アルゴリズム1.1のオーダー

- 最初（ステップ1）と最後（ステップ3）に1文づつ実行 → O(1)
- 外側ループ（ステップ2）は n-1 回 繰り返す
- 内側ループ（ステップ2-1）は平均 n/2 回繰り返す
- ループ内（ステップ2-1-1）は高々3文実行 → O(1)
  - 利益の計算
  - 利益最大値と利益の比較
  - 利益最大値の更新
- T(n) = 1 + (n-1) × n/2 × 1 + 1 = $O(n^2)$

アルゴリズム1-2（引き算回数を削減）

1. 利益最高値 を 0 に初期化する.
2. 買月 を 0 から n-2 まで変えながら以下の処理を繰り返す.
   2-1. 売最高値 を 買月の株価とする.
   2-2. 売月 を 買月+1 から n-1 まで変えながら以下の処理を繰り返す.
      2-2-1. 売月の株価 > 売最高値 ならば 売最高値 を 売月の株価で更新する.
   2-3. 利益 を 売最高値 – 買月の株価とする.
   2-4. 利益 > 利益最高値 ならば 利益最大値 を 利益 で更新する.
3. 利益最大値を返す.

二重ループ内で
引き算をしない

アルゴリズム1-3(売月を基準に)

1. 利益最高値 を 0 に初期化する.
2. 売月を 1 から n-1 まで変えながら以下の処理を繰り返す.
   2-1. 買最安値 を売月の株価とする.
   2-2. 買月を 0 から売月-1 まで変えながら以下の処理を繰り返す.
      2-2-1. 買月の株価 < 買最安値ならば買最安値 を買月の株価で更新する.
   2-3. 利益 を 売月の株価 –買最安値 とする.
   2-4. 利益 > 利益最高値 ならば利益最大値を利益で更新する.
3. 利益最大値を返す.

ループ変数を入れ替える

もっと早くできないのか？

- アルゴリズム1.1は正しい．しかし，・・・
- 総当たり処理をしている限り，O(n^2)
  - アルゴリズム1.1
  - アルゴリズム1.2, 1.3（改良版）
- 利用できそうな，うまい性質は無いか？
- アルゴリズム1.4
  - 最安値を保持する変数を導入
    - それまでの最安値を保持
  - 各売月に対して，それまでの最安値を計算
    - それまでの最安値よりも安い株価の場合，最安値を更新　→ O(1)

アルゴリズム1.4

1. 利益最大値 を 0 に初期化する.
2. 最安値 を 月0 の株価で初期化する.
3. 売月 を 1 から n-1 まで変えながら以下の処理を繰り返す.
   - 3-1. 利益 を 売月の株価 ― 最安値 とする.
   - 3-2. 利益 > 利益最大値 ならば 利益最大値 を利益で更新する.
   - 3-3. 売月の株価 < 最安値 ならば 最安値 を 売月の株価 で更新する.
4. 利益最大値を返す.

アルゴリズム1.4のオーダー

- 最初に2文（ステップ1～2）
  → O(1)
- 外側ループ（ステップ3）は n-1 回 繰り返す
- 外側ループ内（ステップ3-1～3-3）は 3文実行
  → O(1)
- 最後に1文（ステップ4）
  → O(1)
- $T(n) = 1 + (n-1) \times 1 + 1 = O(n)$

アルゴリズム1.4の威力

- 試算の仮定
  - データ数
    - 1日8時間, 1分毎のデータ → 480データ/日
    - 1年365日分の株価データ → 175,200 データ/年
  - 使用コンピュータ:
    - 1秒間に1億個の命令を実行
- O(n)のアルゴリズム

  → 175200 / 10^8 = 0.001752 [秒]
- O(n^2)のアルゴリズム

  → 175200^2 / 10^8 = 306.95[秒] = 5[分]

演習問題
問1：n次多項式の計算

- $P(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$
  - 係数 $a_n, a_{n-1}, \dots, a_0$ とx の値x_0 を入力して，$P(x_0)$を効率よく計算するアルゴリズムを考えよ．

律儀な方法

アルゴリズム

1. x_0 を入力
2. $a_n, a_{(n-1)}, \ldots, a_0$ を入力
3. $P(x_0)$ を 0とする
4. i を 0 から n まで変えながら以下の処理を繰り返す.
    - 4-1. a_i に x_0 を i 回掛ける.
    - 4-2. 上記の結果を$P(x_0)$ に加算する.
5. $P(x_0)$ を返す.

オーダー評価

- ステップ1 → O(1)
- ステップ2 → O(n)
- ステップ3 → O(1)
- ステップ4 n+1回の繰り返し
- ステップ4-1 → O(n)

$i \leqq n$

- ステップ4-2 → O(1)
- ステップ5 → O(1)

$$1+n+1+(n+1)\times(n+1)+1$$

$$\rightarrow O(n^2)$$

Hornerの方法

アルゴリズム

1. x_0 を入力
2. a_n を入力
3. P(x_0) を a_n とする
4. i を n-1 から 0 まで変えながら以下の処理を繰り返す.
   - 4-1. a_iを入力する.
   - 4-2. P(x_0) に x_0 を 1 回掛けてa_iを加える.
5. P(x_0) を返す.

オーダー評価

- ステップ1 → O(1)
- ステップ2 → O(1)
- ステップ3 → O(1)
- ステップ4 n回の繰り返し
- ステップ4-1 → O(1)
- ステップ4-2 → O(1)
- ステップ5 → O(1)

$$\boxed{1+1+1+n\times(1+1)+1 \rightarrow O(n)}$$

演習問題
問2：区間和の最大値

n個の数値データが配列a[]に蓄えられているとする．0≦i≦j＜nなる区間[i, j]に対する区間和を，a[i]＋a[i+1]＋･･＋a[j] として定義する．区間和が最大になるような区間を求めるアルゴリズムを示せ．ただし，配列には正負の要素が混在しているものとする．

区間和の最大値を求めるための工夫

- 0≦i≦j＜nなる区間[i, j]に対する区間和
  - a[i]＋a[i+1]＋・・＋a[j]
  - T[j] – T[i]
- 区間[0,i]に対する区間和
  - T[i] ＝ a[0]＋a[1]＋・・＋a[i]
- T[i] の値を計算して配列に格納する.
- T[j]-T[i]の最大値を求めれば良い.
- 株取引のアルゴリズムを応用可能
  - アルゴリズム1.1
  - アルゴリズム1.4 → O(n)で計算可能

T[i] =
T[i-1] + a[i]

今日のポイント

- アルゴリズムの良し悪しを，具体例で見た．
- 株式売買による利益計算
  - アルゴリズム1.1〜1.3 → $O(n^2)$
  - アルゴリズム1.4 → $O(n)$
- n次多項式の計算
  - 律儀な方法 → $O(n^2)$
  - Hornerの方法 → $O(n)$
- $O(n^2)$と$O(n)$は，決定的に違う．
  - 工夫することで，処理が劇的に速くなることがある．
  - アルゴリズムは，効果的な工夫の宝庫