The Practice of Programming
勉強会ノート

このドキュメントは、Brian W. Kernighan と Rob Pike の著書である「The Practice of Programming」の第 1 章と第 2 章を、新卒新人を中心として行った勉強会向けに作成したものです。

1999 年 2 月に米国で発売になった「The Practice of Programming」に書かれている内容は、全くの初心者に非常に難しく、一方、経験を積んでいるソフトウェア・エンジニアにとっては当たり前のことが書かれています。

したがって、経験があるソフトウェア・エンジニアが中心となって勉強会を開催する場合のサブノートとして使用されるように、勉強会参加者に対して問い掛ける質問を中心にまとめられています。

1 章と 2 章だけでなく、すべての章で有益なことが書かれていますので、是非全体を読んでください。初心者には分かり難い部分も多いですが、その場合には、職場の経験ある先輩に聞いてください。でも、正しく説明出来る人は多くないと思います。「The Practice of Programming」に書かれている内容がすべて説明できるようになれば、一人前のソフトウェア・エンジニアと言っても過言ではないでしょう。

2000 年 5 月 14 日　柴田 芳樹

バージョン 0.29
2006 年 1 月 23 日
作成 柴田芳樹

1 Style - スタイル

1.1 Names

Use descritive names for globals, short names for locals.

(Q) グローバルな物には、なぜ説明的な名前を付ける必要があるのですか？

(Q) Naming convention に対してどうすることが重要ですか？

(Q) Naming convention はどのような効果をもたらしますか？

(Q) C++言語の Namespace および Java 言語の Package とはどのような機能ですか？

Be consistent.

(Q) 何について、首尾一貫しなさいと説明していますか？

(Q) 改善されたクラス UserQueue には、まだ問題があるようですが、さらに改善してみて下さい。

Use active names for functions.

(Q) 関数名を能動態の動詞で始めるメリットは何ですか？

(Q) checkoctal()と言う名前からだけではどのような関数だと、想像されてしまうと思われますか？

Be accurate.

(Q) 何が正確でありなさいとここでは説明していますか？

(Q) 正確で無いと、どうなると述べていますか？また、同じような経験をしたことはありますか？

Exercise 1-1

(Hint) C 言語では boolean の true および false に相当する値は何ですか？

(Hint) 変数 not_eof を日本語で読んでみてください。

Exercise 1-2

(Hint) Use active names for functions を参照

Exercise 1-3

(Q) この練習問題は、何を意図していると思いますか？

1.2 Expressions and Statements

Indent to show structure.

(Q) 最後に示されているコードの改善点は、なぜ好ましいと説明していますか？

Use the natural form for expressions.

(Q) 悪いコードと改善されたコードを声を出して読んでみてください。

Parenthesize to resolve ambiguity.

(Q) C 言語のすべてのオペレータの優先順位を正確に言えますか？言語仕様書を見ないで、言える人があなたの回りに何人いると思いますか？

Break up complex expressions.

(Q) ここで示されている悪いコードと改善されたコードのどちらが、コンパイルされた時に命令サイズが短くなると思いますか？

Be clear.

(Q) 明瞭性（Clarity）と簡潔性(Brevity)の関係については、どのように説明されていますか？

Be careful with side effects

(Q) C言語およびC++言語では、++の副作用の評価順番が保証されていないと書かれていますが、Java言語ではどうなっていますか？調べてみて下さい。

Exercise 1-4

(Hint) 1番目は、Use the natural form for expressionsを参照

(Hint) 残りは、Break up comples expressionsとBe clearを参照

Exercise 1-5

(Hint) Be careful with side effects参照

Exercise 1-6

(Hint) Be careful with side effects参照

1.3 Consistency and Idioms

Use a consistent indentation and brace style.

(Q) indentationとbraceのスタイルについては、どのようにしなさいと説明してありますか？

(Q) 他の人が書いたプログラムを修正する場合には、スタイルについては、どのようにしなさいと説明されていますか？また、その理由は何ですか？

Use idioms for consistency.

(Q) イデオムを使うメリットは何ですか？

Use else-ifs for multi-way decisions.

Exercise 1-7

(Hint) どれも簡単な問題ですね。

Exercise 1-8

(Hint) 簡単な問題ですね。

1.4 Function Macros

Avoid function macros.

(Q) 関数マクロを使用した場合の問題点は何ですか？

Parenthesize the macro body and arguments

(Q) C++ではinlineを使用しなさいとありますが、関数マクロと比べるとどのような利点があるのですか？

Exercise 1-9

(Hint) Parenthesize the macro body and argumentsを参照。

1.5 Magic Numbers

マジックナンバーとは、プログラムの中に現れる、定数、配列のサイズ、文字の位置り、変換率あるいはその他の数値定数です。

Give names to magic numbers.

(Q) マジックナンバーに名前を付けて使用する利点は何ですか？

Define numbers as constants, not macros.

(Q) マジックナンバーを定義するのに#define 文を使用しない理由をどのように説明していますか？

Use character constants, not integers.

Use the language to calculate the size of an object.

(Q) 使用する配列等のサイズを扱う際に、どのようなプログラムを書くことが重要だと説明していますか？

(Q) 関数マクロの有効な例が述べてありますが、C 言語および C++言語では、それらを普通の関数で実現することは可能ですか？

(Q) Java 言語での配列は、C/C++言語での配列とどのように異なりますか？

Exercise 1-10

(Hint)

FT2METER:	*1 フィートが何メートルか*
METER2FT :	*1 メートルが何フィートか*
MI2FT :	*1 マイルが何フィートか*
MI2KM :	*1 マイルが何キロメートルか*
SQMI2SQKM:	*1 平方マイルが何平方キロメートルか*

1.6 Comments

Don't belabor the obvious.

Don't comment bad code, rewrite it.

Don't contradict the code.

Clarify, don't confuse.

(Q) コメントの目的は、何であると最後に説明していますか？

(Q) コメントを書く前に、何をすることが重要だと最後に述べていますか？

Exercise 1-11

(Hint) 1 番目と 2 番目は、Don't contradict the code 参照。

(Hint) write_message()関数は、何をする関数ですか？

修正歴：

2006 年 1 月 23 日　　一般公開用に、途中の修正歴と翻訳部分を削除

2 Algorithms and Data Structures - アルゴリズムとデータ構造

2.1 Searching

説明ポイント 1　C/C++言語でのコンパイル時に配列が初期化されることと、Java では実行時に初期化されることの違いを説明して下さい。

説明ポイント 2　配列中の要素数を関数に渡す方法が２通り説明されていますが、そのどちらの方法が良いと思うか説明して下さい。

説明ポイント 3　char *array[], char **array のメモリー上のイメージを、ホワイトボードを使用して説明してください。
関数の引数として渡す場合には、何故どちらも同じ意味になるかも説明して下さい。

説明ポイント 4　sequential search 版の lookup 関数の処理を説明して下さい。

説明ポイント 5　sequential search を簡潔に説明して下さい。

説明ポイント 6　strchr, strstr とはどのような関数かを説明してください。また、それらの関数と sequential search との関係を説明して下さい。

説明ポイント 7　sequential search が linear search とも呼ばれる理由を説明して下さい。

説明ポイント 8　Java の String クラスの indexOf メソッドと C++の find についてどのようなものか調べて説明して下さい。

説明ポイント 9　binary search を簡潔に説明して下さい。

説明ポイント 10 binary search を行うためのデータに関する二つの条件を説明して下さい。

説明ポイント 11 binary search 版の lookup 関数の処理を、ホワイトボードを使用して説明してください。データが見つからない場合に no match になるメカニズムも説明して下さい。

説明ポイント 12 binary search における比較回数は、約 $\log_2 N$ 回ですが、sequential search と binary search のどちらを選択するかのポイントを説明して下さい。

2.2 Sorting

説明ポイント 13 quicksort の基本的なアルゴリズムの考え方を説明して下さい。

説明ポイント 14 insertion sort と bubble sort を調べて、そられの基本的なアルゴリズムを説明して下さい。

説明ポイント 15 本文中のソースコードと図を用いて、ホワイトボードを使って quicksort の動きを説明して下さい。

説明ポイント 16 quicksort の処理時間がデータ量を n とすると $n\log_2 n$ に比例する場合を説明してください。

説明ポイント 17 quicksort の処理時間がデータ量を n とすると n^2 に比例するようなケースを説明して下さい。

2.3 Libraries

説明ポイント 18 C ライブラリーの qsort 関数のプロトタイプ宣言を示して、パラメータの意味を説明して下さい。

説明ポイント 19 p.35 にある scmp 関数を同じページの図を用いて説明して下さい。

説明ポイント 20 p.36 にある integer 用の scmp 関数を説明して下さい。

説明ポイント 21 ? return v1-v2 の問題点を説明して下さい。

説明ポイント 22 ANSI C の bsearch 関数のプロトタイプ宣言を示して、パラメータの意味を説明して下さい。

説明ポイント 23 p.36 の書き直された lookup 関数と p.37 の nvcmp 関数を説明して下さい。

説明ポイント 24 C++の sort について、どのようなものか調べて説明して下さい。

Exercise 2-1

説明ポイント　再帰呼び出しを使用しないQuicksortプログラム(C言語)を作成して、実際にデータを用いてソートが正しく行われることを確認して下さい。ソースコードを印刷して配布し、説明を行って下さい。

2.4 A Java Quicksort

説明ポイント25 Java言語でinterfaceを使用する理由を説明してください。

説明ポイント26 interface Cmpを説明してください。

説明ポイント27 class Icmpを説明してください。

説明ポイント28 class Scmpを説明してください。

説明ポイント29 Quicksort.sortを説明してください。

説明ポイント30 クラスQuicksortをテキストのコードをもとに実装して、テストしてください。また、そのソースコード（テストコードも含む）を配布して説明して下さい。

Exercise 2-2

説明ポイント　特定の型のデータのみを取り扱うQuicksortクラスを作成して、type conversionのパフォーマンスを測定して、テキストのQuicksortクラスと比較してみてください。ソースコードを配布してコードの説明、測定結果の説明を行って下さい。

2.5 O-Notation

説明ポイント31 O-notationとは何かを説明して下さい。

説明ポイント32 worst-caseな振る舞いとexpectedな振る舞いとは何か説明して下さい。

説明ポイント33 quick sortの振る舞いとしての$O(n^2)$の可能性がゼロになるような実装をする為には、何か重要だと説明されていますか。

説明ポイント34 全くランダムなデータに対して、テキストで示されているquicksortを実行した際に、$O(n^2)$になる確率は、高いですか？

説明ポイント35 行列の掛け算が$O(n^3)$になることを説明して下さい。

説明ポイント36 $O(2^n)$になる計算例を示して説明して下さい。

説明ポイント37 “Traveling Salesman Problem”とは、どのような問題ですか？

Exercise 2-3

説明ポイント　quick sortがworst-caseな振る舞いをすると思われるデータとランダムなデータを用いて、自分が使用している環境が提供しているquick sortのライブラリーを用いてパフォーマンスの比較を行ってください。比較に用いたプログラムのソースコードを配布して、コードの説明と測定結果の説明を行って下さい。

Exercise 2-4

説明ポイント　実装してプログラムのソースコードを配布して、コードの説明とそのプログラムがどの程度「遅い」のかを説明して下さい。

2.6 Growing Arrays

説明ポイント38 ソート済みの配列にn個の要素を挿入する操作が、$O(n^2)$であることを説明して下さい。

説明ポイント39 動的に増加する要素を処理するためのJava言語およびC++言語のクラスライブライーは、それぞれ何ですか？

説明ポイント40 realloc関数の機能を説明して下さい。

説明ポイント41 addname関数の処理をソースコードに沿って説明してください。

説明ポイント42 配列のサイズを増加させる際に、２倍のサイズにしている理由を説明してください。

説明ポイント 43 p.43 の？のコードは、何が問題かを説明してください。

説明ポイント 44 配列のサイズの初期値を NVINIT=1 としている理由を説明して下さい。

説明ポイント 45 realloc の戻り値をキャストしている理由を説明して下さい。

説明ポイント 46 配列の要素を削除する場合、何を考える必要があると述べていますか？

説明ポイント 47 memcopy 関数と memmove 関数の機能を説明してください。

説明ポイント 48 なぜ memmove 関数を常に使いなさいと説明されていますか。

説明ポイント 49 delname 関数の処理をソースコードに沿って説明してください。

説明ポイント 50 要素を削除する他の方法としてどんな方法があると述べられていますか。

説明ポイント 51 配列は、どのような場合に効果的であり、どのような場合には他の方法を検討すべきだと述べられていますか。

Exercise 2-5

説明ポイント　設問の意味をまず説明してから、回答を説明してください。

Exercise 2-6

説明ポイント　コードを作成して、配布して説明してください。また、このような実装の変更が、addname 関数と delname 関数を使用しているプログラムの他の部分に、どの程度影響を与えるかを説明して下さい。

2.7 Lists

説明ポイント 52 singly-linked list とは何ですか。

説明ポイント 53 配列とリストは、どのような点が異なると説明していますか。

説明ポイント 54 リストと配列は、それぞれどのような場合に使用すると説明されていますか。

説明ポイント 55 リストにはどのような操作があると説明されていますか。

説明ポイント 56 struct Nameval と newitem 関数を説明して下さい。

説明ポイント 57 addfront 関数とその使用例のコードを説明して下さい。

説明ポイント 58 addend 関数を説明して下さい。

説明ポイント 59 addend 関数の操作を $O(1)$ にするには、どうする必要があると述べられていますか。

説明ポイント 60 lookup 関数を説明して下さい。

説明ポイント 61 リストがソートされていても、binary search が使用されない理由は何ですか。

説明ポイント 62 apply 関数を説明して下さい。

説明ポイント 63 printnv 関数とその使用例を説明して下さい。

説明ポイント 64 inccounter 関数とその使用例を説明して下さい。

説明ポイント 65 freeall 関数を説明して下さい。

説明ポイント 66 freeall では、listp->name が free されていません。メモリー割り当てと解放について、テキストでは何を上手く行わないとバグの原因になると説明していますか。

説明ポイント 67 delitem 関数を説明して下さい。

説明ポイント 68 doubly-linked list と何かを説明して下さい。

説明ポイント 69 list pointer と data を別々に割り当てる方法だと、何が可能だと説明されていますか。

説明ポイント 70 list は、どのようなデータを取り扱うのに向いていると説明されています。

説明ポイント 71 どのような場合に tree や hash table が向いていると述べられていますか。

Exercise 2-7

説明ポイント　以下の関数は、C 言語を用いて作成し、テキストで用意されている関数を出来る限り利用して実装して下さい。実装したコードを印刷して配布し、説明して下さい。

説明ポイント　copy は、リストをコピーして別のリストとして返す関数として実装して下さい。

```
Nameval *copy(Nameval *listp);
```

説明ポイント　merge は、二つのリストを引数としてもらい、それら二つのリストをマージして、内容（name）の重複が無い一つのリストを作成して返す関数として実装して下さい。

```
Nameval *merge(Nameval *listp1, Nameval *listp2);
```

説明ポイント　split は、リストの先頭からの index(先頭が 0)で指定される位置でリストを分割して、分割された後を返す関数とし実装して下さい。

```
Nameval *split(Nameval *listp1, int index);
```

説明ポイント　insert は、リストの先頭からの index(先頭が 0)で指定される位置に指定された item を挿入し、そのリストを返す関数とし実装して下さい。

```
Nameval *insert(Nameval *listp, int index,  Nameval *item);
```

Exercise 2-8

説明ポイント　以下の関数を C 言語を使用して、引数 listp で渡されるリストを逆順に並べ替え、並び替えた後の先頭のリストを返して下さい。再帰呼び出しと非再帰呼び出しの二通りの方法で実装して下さい。テキストの指示にあるように、新たに list の要素を生成してはいけません。実装したコードを印刷して配布し、説明して下さい。

```
Nameval *reverse(Nameval *listp);
```

Exercise 2-9

説明ポイント　リストを実現するための型 List をテキストの問題の指示に従って、作成して下さい。リストを操作するための関数や操作を作成する必要はありません。また、言語による長所・短所を説明して下さい。

Exercise 2-10

説明ポイント　リストを操作する関数（Exercise 2-7 の copy 関数だけで良い）が正しく動作しているかを確かめるためのテストプログラムを作成してください。対象とするリストは、テキストで使用している Nameval のリストとして下さい。実装したコードを印刷して配布し、説明して下さい。

2.8 Trees

説明ポイント 72　テキストで説明に使われている binary search tree はどのような物であるかを Nameval の構造体と一緒に説明して下さい。

説明ポイント 73　リストや配列で $O(N)$ を要する操作が、木構造では $O(\log N)$ しか要しないと説明されていますが、操作を 2 つ以上選んで、具体的な例を説明して下さい。

説明ポイント 74　insert 関数の処理をホワイトボートを使って視覚的に説明して下さい

説明ポイント 75　木構造用の insert 関数では、要素の重複をチェックしています。リストの insert 関数では、要素の重複をなぜチェックしなかったと説明されていますか。また、何故、木構造用の insert 関数ではチェックしているのですか。

説明ポイント 76　balanced されている木構造とは、どのような木構造ですか。

説明ポイント 77　balanced されている木構造の利点は、何であると説明されていますか。

説明ポイント 78　木構造にどのように要素を追加していくと、unbalanced になると説明されていますか。具体的な例を挙げて説明して下さい。

説明ポイント 79　木構造が balanced されていることを保証する実装をすることは難しいと書かれていますが、何故ですか。具体的な例を挙げて説明して下さい。

説明ポイント 80　lookup 関数を説明して下さい。

説明ポイント 81　tail recursion とは何ですか。

説明ポイント 82 nrlookup 関数を説明して下さい。

説明ポイント 83 *in-order* な木構造の探索を、ホワイトボードを使用して視覚的に説明して下さい。その際に、applyinorder 関数も説明してください。

説明ポイント 84 木構造を使用してどのようにソートが出来ると説明されていますか。

説明ポイント 85 *post-order* な木構造の探索を、ホワイトボードを使用して視覚的に説明して下さい。その際に、applypostorder 関数も説明してください。

説明ポイント 86 *post-order* な木構造の探索は、どのような場合に使用されると説明されていますか。

説明ポイント 87 *pre-order* な木構造の探索を、ホワイトボードを使用して視覚的に説明して下さい。

説明ポイント 88 *parse tree* とは、どのような木構造か説明して下さい。

Exercise 2-11

説明ポイント　プログラムを作成して、lookup 関数と nrlookup 関数のパフォーマンスを測定して、再帰呼び出しのオーバーヘッドを測定して下さい。実装したコードを印刷して配布し、測定結果と一緒に説明して下さい。

Exercise 2-12

説明ポイント　プログラムを作成して、設問にある質問を回答するための測定を行って下さい。実装したコードを印刷して配布し、測定結果と一緒に説明して下さい。

Exercise 2-13

説明ポイント　テキストにある、insert 関数、lookup 関数、applyinorder 関数、applypostorder 関数の処理が正しいかを確認するテストプログラムを作成して下さい。実装したコードを印刷して配布して、説明して下さい。

2.9 Hash Tables

説明ポイント 89 ハッシュテーブルは、コンピュータサイエンスの重要な発明の一つであると述べられていますが、動的データを保存したり検索したりするための効率的な構造を作り出す為に、何を組み合わせていると説明されていますか。

説明ポイント 90 シンボルテーブルとはどのような物ですか。

説明ポイント 91 hash fucntion、hash value は何であると説明されていますか。

説明ポイント 92 struct Nameval および symtab を p.55 の中の図を用いて説明して下さい。

説明ポイント 93 lookup 関数の動作を説明して下さい。

説明ポイント 94 ハシュテーブルを実現するための配列の大きさは、どの程度であると説明されていますか。

説明ポイント 95 hash 関数について、何を決めなければならないと説明されていますか。その際、hash 関数の振る舞いがどうであるべきだと説明されていますか。

説明ポイント 96 文字列用の hash 関数のサンプルコードを説明して下さい。

説明ポイント 97 ハッシュテーブル用の配列のサイズを素数とするのが賢いと説明されていますが、なぜそうなのか説明して下さい。

説明ポイント 98 初期の Java のリリースに含まれいた String 用の hash 関数はどのような実装がされていて、どのような問題があったと説明されていますか。

説明ポイント 99 JDK1.4 に含まれる String クラスの hashCode メソッド部分をプリントアウトして配布して説明して下さい。

説明ポイント 100 hash 関数は、どのよう文字列に対してテストする必要があると述べられていますか。また、その理由は何ですか。

説明ポイント 101 物質の三次元座標をハッシュすることにより物質全部をシュミレーションする際のメモリー容量を減らすことがなぜ出来るのか説明してください。

説明ポイント 102 Gerard Holzmann とは誰か調べて、説明して下さい。

説明ポイント 103 Gerard Holzmann の Supertrace プログラムとハッシュテーブルの関係を

説明してください。

説明ポイント 104 テキストの最後のパラグラフでは、ハッシュテーブルをどのようにまとめとして説明しているか説明してください。

Exercise 2-14

説明ポイント　テキストの文字列用の hash 関数が上手く機能しないデータセットを作り、プログラムにより確認してください。

説明ポイント　異なる NHASH の値に対応する振る舞いが悪くなるデータセットを見つけるのは容易ですか。

Exercise 2-15

説明ポイント　テキストにあるハッシュテーブルで課題のプログラムを作成してください。プログラムリストを配布して説明して下さい。

Exercise 2-16

説明ポイント　プログラムを作成・テストして、プログラムリストを配布して説明して下さい。

Exercise 2-17

説明ポイント　二次元座標をハッシュするハッシュ関数を作成して、実際にテストプログラムを作成して、どのようにそのハッシュ関数が振る舞うかを確認して下さい。プログラムリストを配布して説明して下さい。

説明ポイント　座標の表現方法が変更（整数から浮動小数点へ、デカルト[平行]座標から極座標へ、あるいは 2 次元から多次元へ）にどの程度容易に対応できるようなハッシュ関数になっているか説明して下さい。

修正歴：

2006 年 1 月 23 日　　一般公開用に、途中の修正歴と翻訳部分を削除