特集：インフォプロのためのプログラミング事例集

インフォプロのためのプログラミングのススメ

前田　朗*

本稿では筆者の経験を事例として示しつつ，インフォプロを対象に仕事でのプログラミングへの向き合いかたを提示する。プログラミングができることで，仕事における課題をコンピュータにより解決すべく取り組んでいける。副次的に身につくプログラミング的思考は，自らが情報システムのありかたを考えるにあたり助けになる。プログラミングは必ずしも難しくなく，簡単なプログラミングであっても，仕事の課題を解決していけることも多い。プログラミング自体に楽しみを見出すことも，周囲やベンダーとのコミュニケーションに役立てることもできる。

キーワード：インフォプロ，プログラミング，事例

1. はじめに

インフォプロに向けて，よりよい仕事を行うための方法としてプログラミングを勧めたい。プログラミングだけが方法ではないが，情報システムに使われるのではなく使いこなす側へ，また仕事上の問題解決を他者に委ねるのではなく自ら解決していく側へと，立ち位置を変えていける。情報の検索・解析・提供は電子情報を抜きに語れず，その基盤がコンピュータである。そして，そのコンピュータはプログラムにより稼働している。つまり，プログラミングは情報利活用の中核に手をかける技術といえる。既存のアプリケーションで，十分に仕事をこなせると考える向きもあろう。しかし，プログラミングで何ができるか理解しきれず，解決できる問題を見落としていないか。そして実際にプログラミングができれば，自力で問題解決にまで至れるかもしれない。また，単調な作業にストレスを感じがちであれば，ぜひプログラミングを勧めたい。知的で気の利いたやりかたで問題を解決していくことや，事業へのより大きな貢献ができることに，楽しみを見出せるはずである。

さて，プログラミングのススメは一般にも言われている。最近の注目ニュースは，小学校における「プログラミング教育」[1]の2020年度からの必修化である。ここでは，システムの仕様からプログラムのコードに落とし込むコーディングのスキルではなく，「プログラミング的思考」の習得に主眼を置いている。つまり，プログラミングが論理的思考の訓練になると目されている。先にも述べたように，プログラミングは情報利活用の中核に手をかける技術といえる。情報と向き合うインフォプロであれば一般よりもなおのこと，仕事でプログラミングを生かせる機会を得ることもできよう。たとえ簡単なプログラムでも十分に実用になる。筆者の経験からしても，他部署が長時間残業で進めていた業務や，「できない」と言われていた懸案を，即興のプログラミングで解決したことがある。うまく状況が当てはまると周囲が驚くほどの成果がでる。

本稿では，筆者の経験を事例として挙げつつ，インフォプロのためのプログラミングの向き合い方を提示する。プログラミングの入門書は数多いが，その学習動機付けについての説明は，今後のIT社会において必要になるといった程度であり，具体例が示されないのが普通のようである。これには職種により状況が異なるため，一般的な説明をしづらい事情があるのかもしれない。本稿では対象をインフォプロに絞ることで，できるだけ事例を示しつつ説明をしていく。

事例を取り扱うにあたり，筆者のプロフィールを簡単に紹介する。プログラミングの受講歴は，大学の図書館情報学課程において，必修であったプログラミング演習と選択で受講したC言語の授業くらいである。本格的なプログラミング学習は，東京大学の図書系事務職員としての20数年に渡る仕事の中で，ほぼ独習といえる形で行ってきた。いままでの担当業務には，図書館の閲覧サービスや資料の管理業務や，図書館情報システムの運用・管理業務などがある。国立情報学研究所への出向では，機関リポジトリ担当を勤めた。本務外には，「図書系職員のためのアプリケーション開発講習会」[2]講師と，専門用語自動抽出システム「言選Web」[3][4]の開発担当の経験がある。主に使用してきたプログラミング言語はPerlであるが，3年前からPythonに切り替えつつある。

2. 難しくないプログラミング

2.1 簡単なプログラムでも役に立つ

プログラミングを勧めるにあたり，まず心理的なハードルとなりがちな，プログラムが難しいという誤解から解いておきたい。

事務仕事の片手間で使うプログラミングであれば，簡便

＊まえだ　あきら　東京大学情報システム部
〒113-0033　東京都文京区本郷7-3-1
E-mail: maeda.akira@mail.u-tokyo.ac.jp
https://orcid.org/0000-0002-4566-8085

（原稿受領　2020.1.17）

に使えるプログラミング言語を選べば十分である。これには，Windows のバッチファイルや Linux のシェルスクリプト，Python や Perl や Ruby といった軽量のスクリプト言語，Microsoft Office 製品などアプリケーションに付属しているマクロ言語，Web ブラウザの動作を記述する JavaScript などがある。

プログラミングには複雑なロジックも必須ではない。簡単な処理は簡単な，複雑な処理には複雑なロジックになるだけである。簡単なロジックの例には，命令文（コマンド）を実行順に羅列しただけのプログラムがある。これならば，プログラミング言語ごとに扱いが異なる変数・オブジェクトといった概念や，制御構文を理解する必要すらない。

江草（2018）[5] が示した Excel でコマンド群を生成する手法が，まさに命令文の羅列の事例である。筆者も同様の手法をよく使用している。先日も，デジタルアーカイブの画像ファイル名に含まれる全角文字をまとめて半角にする処理を行った。このような場合，本来は繰り返し処理（ループ構文）にするのが，プログラムが短くなりスマートかもしれない。しかし，命令文の羅列のほうがプログラムを検証しやすいこと，プログラムを簡単に分割実行できるなど，使い勝手がよいところがある。

テキストファイル同士のマッチングや，正規表現を使った条件付きの文字列の置換，タブ区切りテキストの項目別の集計なども難しくない。筆者はよく Perl で 10 行から数十行程度の即興プログラミングで日常の仕事をこなしてきた。この程度でも作業時間の短縮や手作業によるミスの削減になり，とても重宝する。

情報解析においても，難しく考え過ぎることはない。情報解析の専門家は難色を示すかもしれないが，よい解析対象データさえ用意できれば，R 言語などで入門書どおりに命令文を逐次投入していくだけでも結果が出る。筆者はその方針で，東京大学の学内図書館室の蔵書評価 [6] や，国内機関リポジトリのコンテンツ解析 [7] をしたことがある。専門家でない場合は，入門書どおりの手法で結果を出したほうが，むしろ周囲に説明し易いようにも思えた。

2.2　インターネットに答えあり

プログラミングで困ったときには，インターネットを頼りにできる。筆者もプログラミングなどの技術に関する知識を記録・共有するためのサービスである Qiita[8] の記事をよく参考にしている。サンプルコードやライブラリといった，すでに公開されているプログラムを見つけベースにできれば，プログラミングの難易度をかなり下げられる。

2.3　「無精」「短気」「傲慢」

プログラミングに取り組むために適正が必要であろうか。筆者は，自身に適正があるかどうかより，「適正があると考えてみる」ことが重要と考えている。それができれば，多少の困難があっても，プログラミングに取り組めていける。

たとえば，地道さや勤勉さはプログラミングに役立ちそうであるが，それだけが正解ではない。むしろ短気で怠惰に過ごしたい傲慢な性格にこそ，プログラミングが向いているとも主張できる。Perl の開発者であるラリー・ウォールは，「プログラミング Perl」[9] の序章において，無精（laziness），短気（impatience），傲慢（hubris）の「プログラマの三大美徳」を挙げている。筆者はこれを自分に都合よく解釈して仕事に臨んでいる。

ラリー・ウォールによる「無精」の説明は「トータルでみたエネルギーの支出を減らすために，多大な努力をするように，あなたをかりたてる性質」とある。筆者の解釈では，楽をするために尽力することである。尽力といっても手作業とプログラミングではその性格が異なる。まず単純労働によるストレスから解放される。また使いどころによっては数分のプログラミングで数時間かかる作業を終わらせられる。それにより生まれた余裕は，次の仕事に取り組むことも，残業を減らすことも，さらなるスキルや知識の習得にあてることもできる。

「短気」については，「コンピュータがサボっているときに感じる怒り」とある。筆者は，仕事が停滞しているときに感じるストレスと解釈している。もちろんプログラミングは万能ではないが，手をこまねいているよりは前向きに解決に取り組めるほうがストレスを感じずに済む。

「傲慢」については，「ゼウスの怒りにふれるほど，プライドが高いこと。また，他人にケチをつけられないようなプログラムを書く（そして維持する）原動力になるもの」とある。筆者は，自作プログラムが他者にとって有用と考えること，またベンダーや研究者との仕事にあたり技術面で引け目を感じず向き合えること，と解釈している。

3.　仕事における楽しみ

3.1　課題解決に向かいあえる

会議の席で課題について最初から「できない」との主張があり，それでよいのかと考えたことはないか。そのときは，プログラミングにより「できない」を「できる」に変えられないか考えてみるとよい。プログラミングのよいところは，実証ができることである。これは口論での平行線を避けることに役立つ。筆者は，会議の席において「できない」という意見を聞きつつ，自分であればプログラミングでどう解決するかを考えるようにしてきた。会議が終わったあとから，部分的にでも解決案となりえるプログラムをさりげなく提示することで，いくつかの提案を受け入れてもらえた覚えがある。

業務の高度化・効率化のために，プログラムにより対処できることを見つけ，あらたな提案を行うこともできる。

筆者の経験になるが，このように課題の解決を繰り返していると，周囲から信用や期待を得られてくる。それは財産といえるであろう。

3.2　自分で考える情報システム

プログラミング学習により，「プログラミング的思考」を鍛えられる。そのスキルの活用で，たとえ実際に使うプ

ログラムを組むことがなくとも，自分なりの情報システムを考えていける。

国立情報学研究所が 2019 年に実施した「大学図書館員のための IT 総合研修」[10] は，リレーショナルデータベースの操作言語である SQL の概要を学んだ上で，数人のグループごとに DB Browser for SQLite[11] で自分たちなりのデータベースを構築する研修であった。SQL はプログラミング言語ではないが，それを組み合わせてシステムの機能を考えるとなれば，プログラミングと同等の思考を行っていることになる。筆者はこの研修におけるグループディスカッションに立ち会ったが，受講生がとても熱心に取り組んでいたことを覚えている。思うに情報に携わる人間にとって，自由に情報システムを考えること自体が楽しいことに違いない。

3.3　ロジック・推理・手品

仕事は人生の多くの時間を費やすものである。可能であれば楽しみを持ちたいのは自然であろう。プログラミングであれば，気の利いたロジックを考え出すことを楽しみにできる。またそれにより周囲が驚いてくれることもある。

プログラムを書くとき特に初学者のときには，ロジックを自分なりに考え出す必要があり，それにはパズルなど知的ゲームと同じ楽しみがある。筆者の最初のプログラミング体験は大学に入学する前で，関数電卓によるロールプレイングゲームの自作であった。とはいえ，迷路を探索してランダムに出現するモンスターを退治するといった操作を，1 行のディスプレイしかない関数電卓で実装した程度のものである。迷路の探索では，操作キャラクターがどの方角を向いているかの管理が必要であった。東西南北の四方があるとして，右回転に割り当てたキーを押すと右 90 度，左回転に割り当てたキーを押すとその逆に回転させたい。いまなら分かりやすさを重視するが，当時は図 1 に示す数値計算による解決法を見つけ，自分で感心していたことを覚えている。

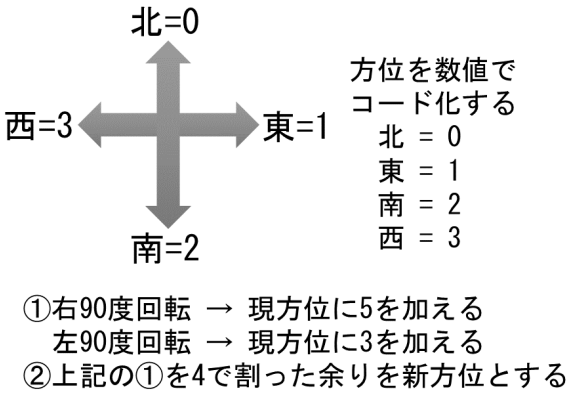

図 1　方角の左右回転

周囲が驚いてくれることも，プログラムを組む動機となりえる。困難と考えられていたこと，そもそも問題と認識してなかったことに，プログラミングで気の利いた解決を示すと，周囲から驚きをもって感心されることがある。筆者にとっては，さながら推理小説における名探偵の推理の披露のようであり，パーティ手品で周囲が驚くことを楽しむのと同種の楽しみであった。プログラミングにおけるバグつぶしは，原因の特定が困難でストレスになりがちであるが，状況証拠を集め原因を特定する推理であると考えれば，それも楽しみと捉えていける。

3.4　ものづくり

プログラミングに慣れてくると，また類似の課題に何度も取り組んでくると，自分で組んだプログラムが動いて当然となり，手ごたえのなさを感じてくるかもしれない。しかし，プログラミングのよいところは，奥が深くこだわれるところにもある。採用する技術の選定や入出力仕様，ドキュメントを含めたプログラムの読みやすさ，処理効率など，手を入れる余地は多くある。自作プログラムの完成度の高まりだけではなく，自身のスキル向上の達成感も得られよう。

筆者は 2002 年から専門用語自動抽出システム「言選 Web」プロジェクトチームにおける開発担当を行っている。このシステムは，日本語・英語・中国語などの文章から，専門用語を抽出し重要度でランキングをするものである。Web アプリケーションとしての提供だけではなく，Perl と Python のモジュールや，Windows 用の応用プログラムの配布を行っている。

このシステムの公開に際しては，中川裕志名誉教授（当時，東京大学情報基盤センター教授）から開発をかなり自由に任せてもらえたこと，勤務時間外での取り組みとしたこともあり，遠慮なくプログラムに手を入れ続けた覚えがある。

このシステムは前身となる JPerl（日本語対応の Perl）のプログラムがあり，了承を得て当時出始めた Perl5 で作り直している。まず機能拡張が容易にできるようモジュール化の設計をし，次に前身のプログラムの内部仕様を明文化した上で，それを参考に読みやすく効率的な処理となるよう考え抜いてみた。職場から帰宅してはプログラムの改訂を繰り返していき，2003 年の公開までには 100 回を越えるバージョンアップを行っていた。

後になって，このシステムの拡張のため，TF*IDF の実装や，N-gram で言語判定を行うオープンソースプログラム TextCat[12] を改変しての組み込み，中国語の GB コード対応などもしている。学生・研究者・企業などユーザーから良い反応をもらえてきたこともあり，それを励みに細かい改修を続けていけた。2018 年には，機能の継ぎ足しで複雑になった仕様の整理も含め，配布モジュールを Python3 で全面的に作り直した。

4. プログラミングのある仕事生活

4.1　足りないパーツを埋める

仕事でプログラムを作成するといっても，本職のプログラマでもなければ，多数の画面をもつような本格的なアプ

リケーションを開発する必要はない。すでにあるシステムでの不足パーツだけを作ればまず事足りる。それには単一の機能だけ提供するツールや，データとシステムの入出力をつなげるためのプログラムがある。

筆者が講師を務める東京大学情報基盤センターの「図書系職員のためのアプリケーション開発講習会」成果一覧[13]では，いくつものアプリケーションを公開している。たとえば，この中にある「junii2 データ診断」[14][15]や「カナフリ」[16]などは，単一の機能を提供するだけのアプリケーションである。「junii2 データ診断」は機関リポジトリシステムに備えて欲しい junii2 ガイドライン準拠のメタデータチェック機能を実装したものである。「カナフリ」は日本語の文にカナの読みを振る Web アプリケーションであるが，実は入出力の部分しかプログラミングをしていない。単にバックエンドで形態素解析システム MeCab[17]を動かしているだけである。

東京大学学術資産等アーカイブズポータル[18]（以降，「アーカイブズポータル」という。）は，東京大学の各部局が個別に公開している貴重書コレクション等を，アイテム単位で横断検索できるサービスである。このアーカイブズポータルのメタデータをジャパンサーチ[19]に CSV 形式で登録することになったが，アーカイブズポータルには CSV 形式の全件出力機能がなかった。そこで，もともと備えていた OAI-PMH プロバイダ機能を使って XML 形式でメタデータを取得し，それをさらにジャパンサーチ登録用の CSV ファイル形式に変換するプログラムを作成した。手作業の部分が残っているが，自作プログラムをパーツとして間に挟むことで，システム間の連携を実現している。

4.2　積み重ねる

プログラミングのスキルは積み重ねがきく。初学者のときは難しく思えたことも，経験を積み重ねることで対応できるようになっていく。過去に経験のないロジックや機能の活用には理解に手間取ることがあるが，一度理解できたことは次回以降の類例に活用できる。過去の自作プログラムが残っていればなおよい。また，類例に取り組むときには，過去の自作プログラムを若干でも洗練させることで，地道ではあるがプログラミングのスキルを向上していける。

先にも事例で示したとおり，筆者は事務仕事の中でプログラミングをしている。ここ 4 年ほどは情報システム部の配属であり使用機会が多いが，東京大学の部局図書館室での事務仕事においても機会があるごとに使用していた。どの図書館室の配属でも重宝したのが，書誌を含むメタデータの処理である。取り組んだことがあるものとして，テキストファイルに含まれる不可視文字の削除，冊子体目録の電子版元原稿からデータベース登録用データへの変換，請求記号による資料一覧のソート，Web API による情報システムからのデータ取得などを覚えている。

4.3　上流工程を押さえる

事務仕事の片手間でプログラミングをする場合は，プログラミングで解決を図るべきか，プログラムをどのような仕様にするか決めるなどかなりの部分を自身の裁量で決められる。初学者のうちは，他者から依頼されたとおりにプログラミングをするのでも学習になる。ただし，いつまでもその立場に留まっていると，便利屋として扱われ続けるだけになってしまう。

システム開発のプロジェクトでは，プログラミングに先立って，クライアントの要望の確認や必要な要件を確定しておくことなどを上流工程という。あらかじめ決められた仕様をもとにプログラムに落とし込むプログラマは特にコーダーとも呼ばれるが，上流工程から開発を担うシステムエンジニアよりも低く評価されがちである。自身の価値を高めるのであれば興味がある課題について，システムの内製においてはクライアントに向き合うベンダーとは立ち位置が異なるが，企画や仕様策定など上流工程に相当することから手掛けるのがよい。組織にはシステムを提案できる人材こそ重要であろう。システム開発における役割を上流工程から下流工程まで体験できることは，ベンダーによるシステム開発の疑似体験ともなりえる。

東京大学学術資産等アーカイブズリンク集[20]は，東京大学の学術資産のコレクションレベルでの一覧 Web サイトである。東京大学デジタルアーカイブ構築事業[21]の本格稼働に先立って，筆者に作成の依頼がきた。この依頼を受けての筆者の提案は，これから本格稼働する事業のアピールを重視した高性能指向のシステムであった。そのコンセプトに基づき，本格的な検索エンジンである Elasticsearch[22]をベースとすることや，情報の補強として Wikipedia 記事データの取り込みをするなどの設計を行い，プログラミングにより独自開発した。このシステムは内部者向けの試作で終わることを想定していたが，他の職員によるデザインテンプレートの修正など改修をかけた上で，2017 年 12 月に一般公開に至っている。

4.4　問題解決力を鍛える

プログラミングは，情報システムにおける問題解決力を鍛えることにつながる。問題解決にあたっては，いかに論理的に原因を特定できるか，それを踏まえて対策案を考えられるかが重要である。行うべきことは，問題の切り分けと，参考情報の確認と理解，仮説の検証である。筆者がみるところ，これは手法の理解より習慣を身に着けるほうが重要である。習慣を身に着けるには意識して繰り返す必要があるが，普通に事務仕事をしていると充分な機会を得ることが難しい。しかし，プログラミングであれば，予期せぬエラーが頻発しがちである。そのエラーの対策に取り組むことが，問題解決のための習慣を身に着ける機会となりえる。

4.5　情報システムの理解に生かす

プログラミングは，その実行環境となるシステムを使い

こなそうとすることに他ならない。例えば，C 言語はオペレーティングシステムの作成にも使われる言語であり，文字列など変数をハードウェアのメモリ操作に近いレベルで取り扱う。Perl は UNIX での利用をメインに開発された言語である。この Perl 言語は，UNIX のテキスト処理コマンド群相当の機能や，UNIX のシステムコール相当の関数などがあり，UNIX で何ができるかを学習できる。LAMP 環境や Web アプリケーションフレームワークにより，実際にアプリケーションを組んでみることも，情報システム開発の理解に役立つ。

情報システムの基礎的な技術を実際に自身でプログラミングしてみることも学習になる。筆者の実例として，すでに 3.4 節で TF*IDF や N-gram のプログラミングをしたことは示した。さらにもうひとつ事例を挙げる。

筆者は Perl による実サービス向けの検索エンジン自作を通して，その技術への理解を深めていけた。その契機は「東京大学学位論文論題データベース」（「東京大学学位論文データベース」[23] の前身）の公開のプロジェクトチームに 1994 年から参加したことである。もともと先輩職員が自作した JPerl による Telnet 端末用の検索エンジンがあり，それをベースに公開することになっていた。しかし，筆者としては当時のマシン環境ではレスポンスが遅いのが気になっていた。そこで Perl のマニュアル本を隅々まで目をとおし，少しでもレスポンスを向上させる方法がないか考え続けた。プロジェクトメンバーとも意見交換しつつ，文字列のパターンマッチの高速化だけでなく，検索データをメモリに入れる手法や，インデックスの利用の検討などを行った。最終的には Web 対応を含めプログラムを完全に作り変え，いくつかの有効な高速化対策を施し 1996 年に一般公開に至っている。その後も開発当時には手を出せなかった，可変長レコードに対しバイナリサーチを行うプログラムを試してみるなど，プログラミングを通して検索エンジンの理解に取り組めた。

筆者は既存のプログラムにあまり頼らず，動作を理解し改変できるものを自作する傾向がある。これを車輪の再発明ととる向きもあろうが，技術の理解には有効であった。学校の数学における定理や公式をいかに導き出すかの授業が，数学的な考え方の理解を深めるのと同じではないかと考えている。

4.6　ベンダーとの仕事に生かす

クライアントとして，ベンダーに過度に依存するのは健全な関係とはいいがたいであろう。共にシステムを開発・運用していくパートナーとしての関係を構築することが望ましい。そのためには 4.5 節に示した情報システムの理解に加え，共通言語として広範な技術用語の理解があるとよい。プログラミングであればその学習の過程で，技術用語の理解を自然と得ることもある。またプログラミングの経験を積むと，ベンダーへの要望に際して難易度を配慮するようになってくる。実際にはベンダーに相談をしてみると想定と異なることも多いが，その配慮はベンダーとの関係構築に重要であると考えている。また運用しているシステムに不具合が発生した際も，プログラミングの経験を活かし，状況からベンダーが必要となる情報を取り揃えて提示するといった対処ができるようになる。

プログラミングのスキルを活かせば，ベンダーにシステムの新規開発を依頼する前に，プロトタイプを作成することもできる。このプロトタイプはベンダーにシステム外注するための予算獲得や，ベンダーに開発システムのイメージを伝えることに役立つ。

筆者はかつて内容要旨や目次情報などを含む商用の書誌サンプルデータを渡され，適当に使ってみるように指示されたことがある。そこで，手の空いたときにプログラミングによりこのサンプルデータを加工し，全文検索システム Namazu[24] に登録してみた。書誌情報中の ISBN をキーに東京大学 OPAC に連携するくらいの工夫しかなかったが学内で評価を受け，「東京大学ブックコンテンツ・データベース」[25]（現在はサービス終了）のシステム外注を含む本サービスの予算がつくことになった。

直近では，アーカイブズポータルのプロトタイプを，中村覚助教らとの 3 人の教職員からなるチームで，オープンソースのディスカバリーシステムである VuFind[26] をベースに作成した。表形式のメタデータをプログラミングで MARCXML 形式に加工し，VuFind の PHP プログラムを直接改変しインターフェイスを変更した。このプロトタイプで得られた評価をもとにシステムの仕様書案を書きあげ，いくつかのベンダーに対しプロトタイプのデモと合わせ，開発するシステムのイメージに齟齬がでないよう説明できた。

5. おわりに

本稿ではインフォプロを対象に，仕事におけるプログラミングの活用を勧めてきた。プログラミングは仕事のための数あるスキルのひとつではあるが，中でも強力なスキルである。筆者の経験上，プログラミングにより「身も蓋もなく」仕事の問題を解決してしまうこともある。少なくとも手持ちのスキルにしておけば，仕事における問題解決の際の選択肢が増えるはずである。

プログラミングを楽しめるかどうかは，個人の気質によるところもあろう。本稿ではプログラミングを楽しみ，ストレスなく仕事をするための考えかたを，できる限り挙げてみた。そのうちのひとつでも参考になれば幸いである。

本稿が読者にとって，プログラミングの学習の動機になることを願っている。

参　考　文　献

1) 小学校プログラミング教育の手引（第二版）. https://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/11/06/1403162_02_1.pdf, (参照 2019-12-30)
2) 前田朗．実用アプリケーションの企画・開発により学ぶ図書館職員向け講習会モデルの考察：「図書系職員のためのアプリケーション開発講習会」を例にして．大学図書館研究．2009，vol.86，p.19-27.

3) “専門用語（キーワード）自動抽出システム”のページ. http://gensen.dl.itc.u-tokyo.ac.jp/, (参照 2019-12-30)
4) 前田朗．キーワード自動抽出システム「言選 Web」．漢字文献情報処理研究．2005，vol.6，p.124-133.
5) 江草由佳．移行しやすく使いやすいデジタルコレクション公開サイト構築の試みー教育図書館貴重資料ジタルコレクション公開準備の経験からー．2018. https://www.slideshare.net/yegusa/20180902-c4ljp2018, (参照 2019-12-30)
6) データマイニングによる図書館蔵書評価の試み. https://mbc.dl.itc.u-tokyo.ac.jp/lecture/librarycollection_datamaining.pdf, (参照 2019-12-30)
7) 尾城孝一．オープンサイエンス概論～オープンサイエンスの推進と機関リポジトリ～．2017. https://www.janul.jp/sites/default/files/2019-10/64_ojiro.pdf, (参照 2019-12-30)
8) Qiita. https://qiita.com/, (参照 2019-12-30)
9) Larry Wall, Randal L. Schwartz（近藤嘉雪訳）．プログラミング Perl．ソフトバンク出版事業部．1993，633p．ISBN 4-89052-384-7
10) 教育研修事業ー大学図書館員のための IT 総合研修. https://www.nii.ac.jp/hrd/ja/it/, (参照 2019-12-30)
11) DB Browser for SQLite. https://sqlitebrowser.org/, (accessed 2020-01-05)
12) TextCat. http://www.let.rug.nl/~vannoord/TextCat/, (accessed 2020-01-07)
13)「図書系職員のためのアプリケーション開発講習会」成果. https://mbc.dl.itc.u-tokyo.ac.jp/products.html, (参照 2020-01-07)
14) junii2 データ診断. https://mbc.dl.itc.u-tokyo.ac.jp/junii2checker/, (参照 2020-01-07)
15) 前田朗．junii2 データ診断．2017. https://www.slideshare.net/genroku/junii2, (参照 2020-01-11)
16) カナフリ．https://mbc.dl.itc.u-tokyo.ac.jp/kanafuri/, (参照 2020-01-07)
17) MeCab: Yet Another Part-of-Speech and Morphological Analyzer. https://taku910.github.io/mecab/, (accessed 2020-01-07)
18) 東京大学学術資産等アーカイブズポータル. https://da.dl.itc.u-tokyo.ac.jp/portal, (参照 2019-12-30)
19) ジャパンサーチ（BETA）. https://jpsearch.go.jp/, (参照 2020-01-06)
20) 東京大学学術資産等アーカイブズリンク集. https://da.dl.itc.u-tokyo.ac.jp/dalink/, (参照 2019-12-30)
21) 東京大学デジタルアーカイブズ構築事業. https://www.lib.u-tokyo.ac.jp/ja/library/contents/archives-top, (参照 2020-01-10)
22) Elasticsearch. https://www.elastic.co/jp/products/elasticsearch, (参照 2020-01-05)
23) 東京大学学位論文データベース. http://gakui.dl.itc.u-tokyo.ac.jp/, (参照 2019-12-30)
24) 全文検索エンジン Namazu. http://www.namazu.org/, (参照 2020-01-05)
25) 茂出木理子, 杉田いづみ, 前田朗．東京大学ブックコンテンツ・データベースの紹介．薬学図書館．1999，44(4) pp.342-344，https://doi.org/10.11291/jpla1956.44.342, (参照 2020-02-23)
26) VuFind. https://vufind.org/vufind/, (accessed 2020-01-05)

Special feature: Programming for Information Professionals. Recommendation of programming for information professionals. Akira MAEDA (Information System Department, The University of Tokyo, 7-3-1, Hongo, Bunkyoku, Tokyo 113-0033, Japan)

Abstract: In this paper, I propose how to approach programming for information professionals, with presenting my experiences as practice. Mastering programming will help you to solve job problems of information professionals by yourself. For thinking about information systems, thinking method and knowledge of information technology through programming experience will help you. Even easy programming, you can solve job problems in some cases. You will find your pleasure in thinking about programming logic, and skill of programming help communication with your companion and system vendors.
Keywords: Information professional / Programming / Best practice