情報学部におけるプログラミング教育 ～過去・現在・未来～

Programming Education in Faculty of Comprehensive Informatics
～Past, Present and Future～

野村 恵美子*1，菅沼 義昇*2，幸谷智紀*1

Emiko NOMURA, Yoshinori SUGANUMA, Tomonori KOUYA

Abstract: Faculty of Comprehensive Informatics in Shizuoka Institute of Science and Technology (SIST, for short), are providing own deep curriculum alignment for learning programming languages at various levels. Our department has been established on the motto, "more practice than theory." Therefore, we let SIST students grow up as standalone software engineers with our unique programming curriculum, which has been sometimes revised to follow changing ICT world since 1991. In this paper, we describe the revision history of our programming education based on memoirs of professors in charge and some figures explaining programming curriculums in corresponding terms.

1. 初めに

第2次世界大戦後に誕生した実用的なノイマン型電子計算機は，当初の予定をはるかに超えるコンピューティング能力の向上を見せ，1980年代のダウンサイジング化により企業・家庭・教育現場に普及した．その結果，1990年代のWebをキラーコンテンツとするインターネットの爆発的拡散が起こり，21世紀初頭にはIT革命と称されるまでにインターネットに接続可能なパーソナルコンピュータ(PC)が全世界的に普及し，現在のICT (Information and Communication Technology)社会を形成するに至った．スマートフォンといえども，UNIXライクなOSカーネルを土台とするPCのアーキテクチャを踏襲したものに過ぎず，スマートフォンの代表的OSであるGoogleのAndroidやAppleのiOS上で動作するアプリケーションソフトウェア（所謂「スマホアプリ」）も，従来のPC用のソフトウェア同様，C/C++，Java等のプログラミング言語を使ってプログラムを人力で開発していく必要がある．このような情報化された社会におけるプログラミングのニーズは増えこそすれ，減ることはなく，動作環境やプログラム言語の変化はあれ，当分は人間がプログラミングのスキルを磨き，目的とするソフトウェアを顧客や世間，自分自身のニーズを勘案しながら地道に構築しなければならないようである．

理論より実践を重要視する教育を掲げて静岡理工科大学が開学したのは1991年(平成3年)4月である．開学時の理工学部知能情報学科は，人間の「知覚機能」をコンピュータの「高速情報処理機能」で実現すべく，プログラミングを通じたコンピュータの取り扱いを実践的に学ぶことを第一に掲げて「プログラミング教育」という土台を持って出発した．そのコンセプトは2017年度から始まった情報学部にも受け継がれており，多様な研究・社会のニーズに沿ったプログラミング教育が各講義(例えば菅沼のテキスト[3]等)，各研究室で行われているが，その基礎はII類理工学基礎科目群，III類専門科目群として実施されてきたプログラミング基礎関係科目で培われている．開学当時より実施されてきたこれらのプログラミング基礎科目は，必ずしも教員の意図通りに展開したわけではなく，四半世紀以上に渡る試行錯誤を経て現在の形式に至っている．新たに情報学部が発足したこの段階で，この試行錯誤の歴史を振り返り，現在の状況について議論しておくことは，未来のプログラミング教育を考えるためにも意義深いことである．

本稿では，まず第2節で本学における情報系学科・学部の変遷について触れる．日本の大学の中では必ずしも長いとはいえない本学の歴史においても，理工学部知能情報学科→同学部情報システム学科→総合情報学部→情報学部という，3回もの看板の書き換え以上の激変を経てきた学科・学部は本学部以外には存在しない．その間，当然，カリキュラムの変更もその都度行われており，プログラミング教育体系も大小の修正を行ってきた．本稿の主題の一つはこのプログラミング関係科目改変の狙いと，その結果について考察することにあるので第3節で触れ，現在の状況を議論する．その議論と現在の情報化社会の動向を踏まえ，未来のプログラミング教育について第4節で議論する．最後に結語を述べる．

2. 情報学部に至るまでの学科・学部の変遷

前述したように，現在の情報学部は，開学当時より3回の名称及び組織変更を行っている．当然その都度，プログラミングをはじめとする教育カリキュラムの改変も行っており，必要に応じて手直し的な小変更も実施している．ここでは学生便覧[1]に記載されている本学の沿革に沿って，現在の情報学部に至る第1期～第4期の大筋を述べる．

第1期・・・1991年4月～1999年3月　理工学部　知能情報学科

静岡理工科大学開学と同時に，理工学部に機械工学科，電子工学科，知能情報学科，物質科学科の4学科が設置され，最初期のコンセプト「理論より実践」に基づいてプログラミング重視のカリキュラムを実施した．教育棟5階にはPC(初期はNEC PC-9801マシン)と，III類の専門情報教育用としてSolarisワークステーションが配置され，最初期のプログラミング教育は主としてこのPCで実施された．その後，Windows 95の登場を経て，1990年代後半からはDOS/VタイプのPCを用いた

2018年1月19日受理
*1 コンピュータシステム学科，*2 本学名誉教授

II類（学部共通

プログラミング基礎

計算機言語基礎	2	1前	必
電子計算機演習	2	1通	必
数値計算入門	2	1後	必

III類（学科固有）

プログラミングの周辺

情報科学概論	2	1前	必
情報管理論	2	2後	S必
計算理論	2	2後	必
コンパイラ	2	3前	選
オペレーティングシステム	2	3前	選
データベース	2	3後	選
計算機応用システム	2	4前	選
情報ネットワーク	2	3後	選
マン・マシン・インタフェース	2	3後	選
論理回路及び演習	2	3後	S必
計算機アーキテクチャ及び演習	3	1後	必

III類（学科固有）

プログラミング

プログラミング言語及び演習	3	1後	必
アルゴリズムとデータ構造	2	2前	選
プログラム論	2	2前	選

計算機科学の分野を通じて応用的にプログラムする

数値解析第1及び演習	3	3前	S必
数値解析第2及び演習	3	3後	選
非数値処理及び演習	3	2前	選
自然言語処理	2	3前	選
パターン認識 I	2	2後	選
パターン認識 II	2	3前	選
人工知能 I	2	2後	A必
人工知能 II	2	3前	A必

図 1: 1991～1995 年入学生のプログラミング関係科目

実習に変更され，学生個人に Note PC を持参させて実習させる形態に変更，現在に至るまで，コンピュータリテラシ，プログラミング教育等，全ての情報処理スキルは学生個人の Note PC を用いて実施されるようになっている．

第 2 期　・・・　1999 年 4 月～2008 年 3 月　理工学部情報システム学科

知能情報学科を改組し，情報システム学科を開設．折からの IT ブームに乗っかる形で定員を 80 名から 140 名へと大幅増．一時は 1 学年 200 名を超えたこともある．インターネット環境と学内ネットワークが整備され，Web ブラウジングのために研究室で寝泊まりする学生も出現した．

プログラミング教育としては曲がり角を迎え，必修科目の習得ができない学生が多数出現し，夏休みに補講するなどして対応した．学力的な難しさに加え，PC のコモディティ化によって，Microsoft Office をはじめとするアプリケーションソフトウェアの利用が重要視されてくるようになり，プログラミングの重要性が認識されづらくなったきらいもある．その後のカリキュラム改正に繋がる過渡期と言える．

第 3 期　・・・2008 年 4 月～2017 年 3 月 総合情報学部（コンピュータシステム学科，人間情報デザイン学科）

第 2 期におけるプログラミング教育の難点を克服すると共に，Web をはじめとする「メディア」を重視したカリキュラム構築の必要性も生じていたため，アドバンストコースとして「Web デザイン特別プログラム」が設置されるとともに，学生の習熟度に応じたプログラミング教育体系の大変更を行い，現在に至っている．理工学部建築学科の新設に伴い，収容定員の削減（140 名→120 名）も実施されたが，実質的な学生数にはあまり影響はなく，実践的なプログラミングスキルを身につけられる学生の割合は低下傾向が続いた．

第 4 期　・・・ 2017 年 4 月～　情報学部（コンピュータシステム学科，情報デザイン学科）

コンピュータシステム学科については第 3 期のカリキュラムを引き続き実施，情報デザイン学科にはメディア関係科目の新設・必修化とプログラミング科目を選択化した．人間の感性に訴える IoT(Internet Of Things)，メディアを媒介とした教育を目指し，特別プログラムは「デザイン志向」の内容へと衣替えする．

プログラミング科目については第 3 期のものを踏襲し，コンピュータシステム学科は大部分を必修化したままになっている．

3. プログラミング教育の過去と現在

前節で第 1 期～第 4 期の学科・学部名称変更とカリキュラム改変のあらましを述べた．本節ではその詳細について，「プログラミング教育体系とその狙い」と「教育効果と反省点」について，プログラミング関係科目のみ抜粋して図示したものと共に述べる．

3.1 第1期 1991～1995年度入学生までのカリキュラム

「理論より実践」というポリシーのもと，当時はまだ非力であったワークステーションやパソコン上でプログラミングスキルを磨くためのカリキュラムが組まれた（図 1）.

プログラミング教育体系とその狙い

この創立当時は理工学部全体としても，プログラミング言語はSolarisワークステーションでFORTRANを使用し，数値計算をベースにプログラミングの基礎を学習していた．プログラミングの入門は学部共通科目の「計算機言語基礎」「電子計算機演習」で，講義と演習形式で実施していた.

知能情報学科では，さらに，学科固有科目で一般的なレベルまでプログラミングを学習して，情報各分野の学習と合わせて上級のプログラミングを身につけるようなカリキュラムを組んでいた.「理論より実践」を体現するためのプログラムは学習すれば誰でもそれなりにできる期待があった.

教育効果及び反省点

しかしながら，当初予想したほど知能情報学科の学生がプログラムを組めるようにはならなかった．むしろ，かなりの程度，できないと言ってよいレベルであり，教える側としても非常に戸惑った．教員の側には，学生の意欲が足りないという認識が一般的であり，自己反省を促すべく厳しく指導すれば出来るようになるはずだという信念を持つ教員が多かった．理工学部の他学科では，さらにその傾向が強かったと思われる．また，理工学部共通科目としての「計算機言語基礎」に対する重点の置き方に温度差があり，知能情報学科ほどプログラミングスキルが重要視されていなかった

3.2 第1期 1996~1998年度入学生までのカリキュラム

コンピュータ教室のリプレースが順調には実現せず，この時期から，普及が始まったWindows95 のノートパソコンを学生が購入して持参する方式を取ることになり，505教室，506教室のPCを廃止した．また，プログラミング言語もFORTRANからCへ変更した.
GUI OSの普及期であり，まだ高校以前にPCのアプリケーションソフトを使ったことのない学生が多数であったため，「コンピュータ入門」（Ⅱ類）を導入して，Microsoft Office による文書作成の教育を行うようになったのもこの時期からである．この時期のプログラミングカリキュラムを図2に示す.

前述したように，理工学部共通科目としての「計算機言語基礎」に対する重点の置き方に温度差があったこと，知能情報学科としては，プログラミング教育を強化してプログラムが作れる学生を育てることに注力しなければならない必要性から，学部共通科目とは別立てのカリキュラムに転換する方向になった.

Ⅱ類（知能情報）

コンピュータ入門	1	1前	必
計算機言語入門	2	1後	選
フォートラン言語	2	3前	選択

プログラミング

プログラミング言語及び演習	3	1後	必
アルゴリズムとデータ構造	2	2前	選
システムプログラミング演習	1	2後	選
ソフトウェア工学	2	3前	S選

プログラミングの周辺

情報科学概論	2	1前	必
計算理論	2	3後	S選
コンパイラ及び演習	3	3後	S選
オペレーティングシステム	2	3前	S選
データベース	2	3後	選
ヒューマンインタフェース	2	4前	選
コンピュータグラフィックス	2	4前	選
システム設計	2	3後	S選
計算機ハードウェア	2	2前	必
計算機アーキテクチャ	2	2後	必

計算機科学の分野を通じて応用的にプログラムする

数値解析第1及び演習	3	3前	S選
数値解析第2及び演習	3	3後	S選
記号処理プログラミング及び演習	3	2後	選
自然言語処理	2	3後	A選
パターン認識1	2	3前	A選
パターン認識2	2	3後	A選
人工知能1	2	3前	A選
人工知能2	2	3後	A選
符号・暗号理論	2	3後	S選
オートマトンと形式言語及び演習	3	3前	S選
モデリングとシミュレーション	2	2後	選

図 2: 1996年～1998年のプログラミング教育カリキュラム

プログラミング教育体系とその狙い

全体としての枠組みには，それほど大きな変更はなかったが，プログラミングを継続的・段階的に学習するようなカリキュラムになった．また前述したように，Windows や Microsoft Office を用いたリテラシー教育も開始された.

理工学部共通のプログラミング科目は選択となったので，1年前期にリテラシーを兼ねて計算機の操作に慣れて，1年後期からプログラミングを学科固有科目で実施する体制になり，知能情報学科としては，1年から3年までの段階的学習で，それなりにできる期待があった.

教育効果及び反省点

著者の一人である野村は，1996年度より1年生のプログラミング言語および演習を担当するようになり，当初からなかなか理解が進まない様子を本学では初めて目の当たりにした．どうすれば幾らかでもプログラミングに対する理解が進むのか，皆目見当がつかず，段階的学習になるような計画とはなってはいたが，前段の知識が定着しないので，担当者としては，益々混迷を深めていた.

積み上げ的学習が成功しないことは，他分野，他学科でもあったようで，教授会でも発言されたと記憶する．まだ「出来ない奴は努力が足りない」という根性論が根強かった.

3.3 第2期 1999~2002年度入学生までのカリキュラム

知能情報学科から「情報システム学科」に再編され，次のような大変化があった.

- 大幅な定員増（80名 → 140名）を，コース増（2コース → 3コース）とそれに伴うコース選択科目増で対応
- 新規教員の採用（著者の一人である幸谷もこの時に採用される）
- 基礎教育室の解体に伴う教員の異動

・ 3つ目のコースは，経営社会系の要素はあったが，ネットワークの一部のような曖昧な位置付けであった．

結果として，知能情報の人工知能的一体感（計算機科学と生命・心理系の一体感）は解体された印象を持った．この時期のプログラミング関係科目を図3に示す．

II類（情報システム）

フォートラン言語	2	3前	選

リテラシー

コンピュータ基礎	1	1前	必

プログラミングの周辺

計算理論	2	3後	C選
コンパイラ及び演習	3	3後	C選
オペレーティングシステム	2	3後	C選
データベース	2	3後	選
ヒューマンインタフェース	2	3後	C選
コンピュータグラフィックス	2	3前	N選
マルチメディア論	2	2前	選
映像と音楽	2	2後	N選
システム設計	2	3前	C選
計算機ハードウェア	2	2前	選
計算機アーキテクチャ	2	1後	選
電子回路基礎	2	2後	C選

プログラミング

プログラミング1	2	1後	必
プログラミング2及び演習	3	2前	必
システムプログラミング演習	1	3前	C選
アルゴリズムとデータ構造	2	2後	選
プログラミング方法論	2	3前	選

計算機科学の分野を通じて応用的にプログラムする

数値解析及び演習	3	3前	C選
自然言語処理	2	4前	選
聴覚情報処理	2	2後	選
視覚情報処理	2	3前	選
GAとニューラルネットワーク	2	2前	選
人工知能	2	2後	選
符号・暗号理論	2	3後	N選
オートマトンと形式言語及び演習	3	2後	C選
モデリングとシミュレーション	2	2後	選

ネットワーク

インターネット言語	2	2後	N選
インターネット論	2	1後	選
コンピュータネットワーク	2	3後	選
ネットワーク管理	2	3前	N選

図 3: 1999 年～2002 年度　情報システム学科プログラミング関係科目

プログラミング教育体系とその狙い

コースの再編と，プログラミング関連分野と計算機科学分野の科目の多少の推移はあったが，プログラミング関係の科目自体は大きな変更はなかった．この時期のプログラミングカリキュラムを図3に示す．

それよりも，プログラムができるようにならない学生の意欲を強く喚起したい意図で，1年後期，2年前期と連接する科目とし，丁寧に導入教育をする一方，評価を厳しく設定した．必修化と再履修により学習しなければならない状況を作ることで，意欲が喚起され，できるようになるはずという期待があった．

教育効果及び反省点

しかしながら，単に必修にしただけでは，問題の解決にはならず，むしろ，再履修者の増加と意欲の喪失（4年次まで再履修になった学生がどうなるのかという不安に対し「先生がなんとかするんでしょ」という会話を聞いた）を招いた．

プログラムができるようになるという課題は，学生の自助努力の範囲を超えていたと考えられる．学生の理解力に沿ったカリキュラム，授業の進度でないと，受け入れられない印象を強く持った．

さらに，学習意欲という面では，極めて意欲に欠ける学生がいると，周囲の学生も努力する意欲を失う傾向にあるように見受けた．そのことから，入門レベルでは少しの努力でわかりそうな易しめの内容に抑え，次のステップへは，進んで継続できそうな学生とそうでない学生を分ける必要を感じた．

3.4　第2期 2003~2007 年度入学生までのカリキュラム

Internet の本格的普及期を迎えたことから，Web 技術の発展に伴い，ネットワーク関連の科目をコースから共通科目化し，ネットワークコースは経営社会分野の方向になった．また，計算機科学系の科目は，プログラミング，Web 関係，CG とマルチメディア方面に転換した．

新入生への手厚いサポートの必要性が叫ばれたことから，コンピュータリテラシ（当初8クラス，コンピュータ基礎とフレッシュマンセミナーの合同講義），プログラミング入門（当初10クラス）教育の少人数化を行ったのもこの時期からである．

プログラミング教育体系とその狙い

1999 カリキュラムの反省から，プログラミング教育は，1年後期の「プログラミング入門」（以下「入門」と表記）に制御構造を中心に実行のプロセスを理解することを中心に学習させ，その成績によって，次に履修できる科目を指導する方針とした（図4）．成績が良い学生（優以上）は「プログラミング2」（以下「2」と表記）を科目指定し，それ以外の学生には「プログラミング1」（以下「1」と表記）の科目を履修指定した．ただし，いずれも選択科目とし，取らないという選択をすることも可能であった．また，「入門」は5～6クラス編成と少人数クラスで開講することになった．

このような到達度クラス編成をカリキュラムと組み合わせるにあたり，考慮した点は次の2点である．

1. 教員の側から，到達度クラス編成に大きな抵抗があることが予想され，不用意に審議の場に出すと実現できなくなる可能性を非常に危惧した．そのため，カリキュラム審議の際は，到達度編成であることをなるべく目立たないようにし，科目的にも到達度別でも同等に見えるように配慮した．また，科目担当も「1」系の科目と「2」系の科目で，偏りがないように配慮した．
2. 「入門」の成績による振り分けが必ずしも適切でない場合がありえることである．不適切であれば，能力以上に難しい科目に割り当てられたり，上位クラスでも十分できるにもかかわらず，下位科目を履修しなければならないこともあり得る．そこで，「入門」以外を選択とし，履修科目を指定はするが取らない選択もあるようにし，また，上位科目を履修したいという申告があり，妥当と判断された場合は受け入れることとした．

教育効果及び反省点

上記の1点目については，実際，否定的に見られたり，履修制限に賛同を得られないことはあった．しかし，継続することで次第に受け入れられるようになり，他にも類似の科目が設定され，「アドバンスト科目」のような言葉も使われるようになった．

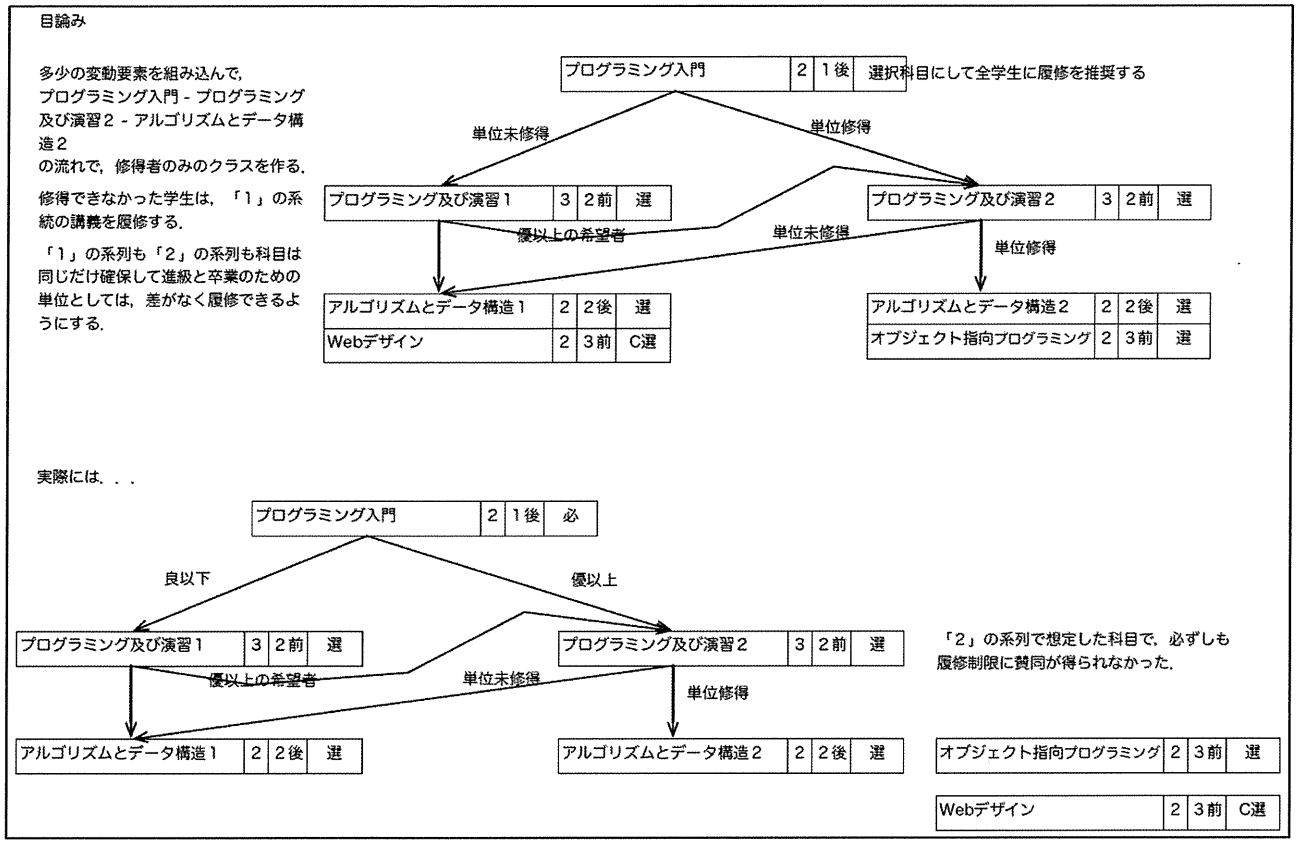

図 4: 2003 年度~2007 年度 プログラミング教育の目論見と実際

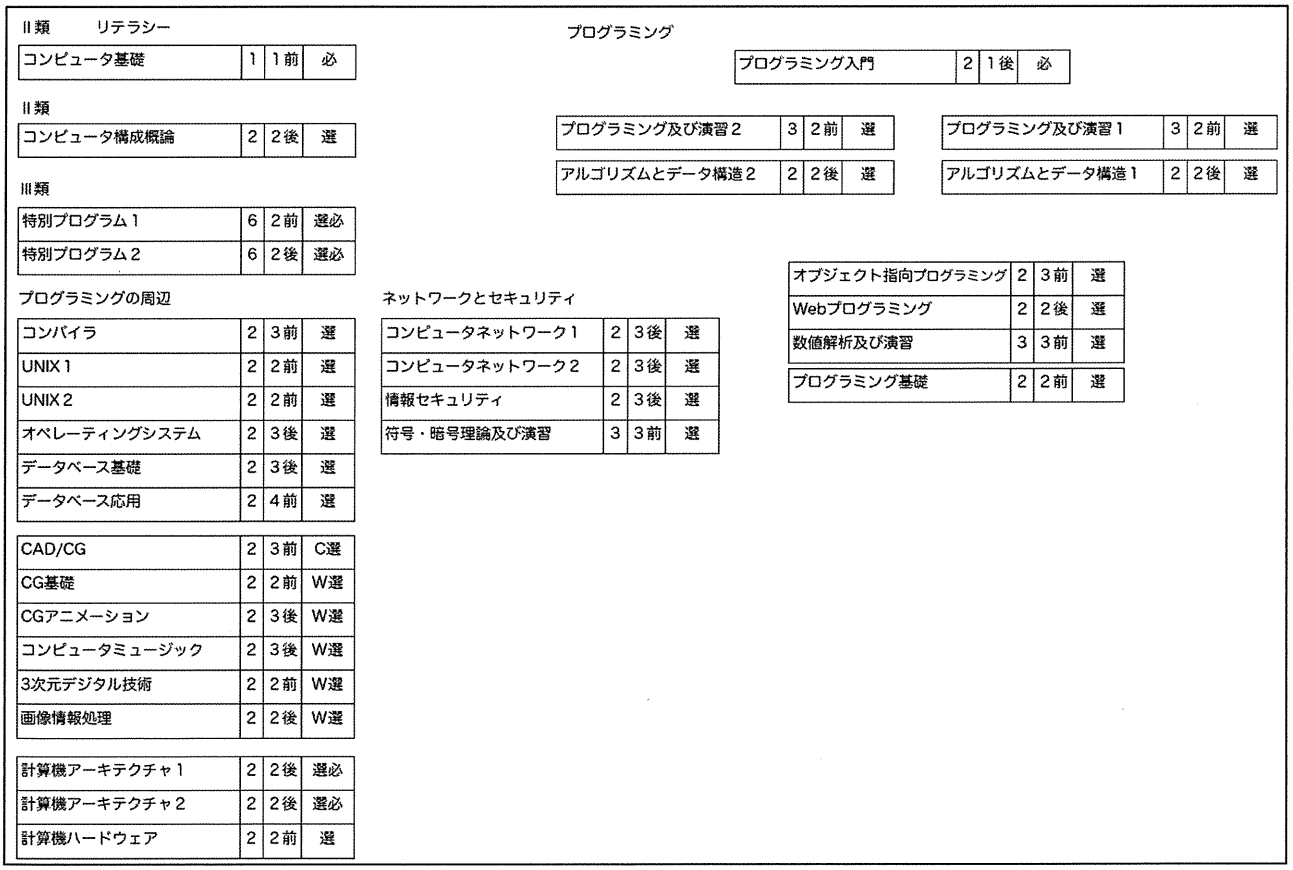

図 5: 2008 年度～2011 年度 総合情報学部コンピュータシステム学科カリキュラム

情報学概論	2	1前	必

特別プログラム1	6	2前	選必
特別プログラム2	6	2後	選必

実践ベンチャービジネス1	10	3集	選必
実践ベンチャービジネス2	10	3集	選必

コンパイラ	2	3前	選
OS (UNIX)	2	2前	選
データベース基礎C	2	2後	選

パターン情報処理	2	3前	選

計算機ハードウェアC	2	2前	選必
計算機アーキテクチャC	2	2後	選必

情報セキュリティC	2	3後	選
コンピュータネットワークC	2	3後	選

プログラミング言語	2	2前	選必

実用プログラミング1	2	2後	選必
実用プログラミング2	2	2後	選必

Webプログラミング	2	3前	選必
データベース応用	2	3前	選必

アニメーションとゲーム2	2	3前	選必

アルゴリズムとデータ構造1	2	2後	選

符号・暗号理論1	2	2後	選
符号・暗号理論2	2	3前	選
数値解析1	2	3前	選
数値解析2	2	3後	選

プログラミング入門+	2	2前	選必
マークアップ言語	2	2前	選必
マクロ言語入門	2	2後	選必
コンピュータ応用	2	2後	選必

アニメーションとゲーム1	2	3前	選必
グラフィックスデザイン	2	3前	選必
プログラミング基礎	2	3前	選必

アルゴリズムとデータ構造2	2	2後	選

図 6: 2012 年度～2016 年度 総合情報学部　コンピュータシステム学科カリキュラム

考慮した２点目について，「２」の系列科目では，極めて意欲に欠ける学生の履修がなくなり，学習意欲は上がったと感じた．その点では，２点目の危惧は心配したほどではなく，比較的うまく機能したと言える．一方，成績の良い方の学生でも，「２」の方の科目を指定されたことで，難しいのではないかという懸念から，履修しないケースが無視できない程度には存在した．履修をどのように推奨すれば良いかが次の課題であったが，名案はなく，手をこまねく期間が長くなってしまった．

プログラミングができるようになったか，については，教育の効果としては判断しにくい．できる学生ができるようになる当たり前の結果しか得られていない．どのような教育内容で，どの程度の効果が得られるものか，見極めるための一層の努力が必要な状態が現在も続いている．

3.5　第 3 期 2008~2011 年度入学生までのカリキュラム

理工学部の一学科では，文理が融合した学際的情報教育が難しいということから，理工学部から独立した「総合情報学部」とし，コンピュータシステム学科と人間情報デザイン学科の２学科体制として発足した．この際の変更点としては下記のものが挙げられる．

- 教職課程（高校一種の情報，数学）の開設
- 「ネットワーク」系のコースが経営社会系へと科目的にもはっきりと転換された．
- 生命・心理・社会系の分野は人間情報デザイン学科の科目となった．
- 人間情報デザイン学科の Web デザインコースに CG，アニメーション，コンピュータミュージック系の科目が整備された．

この時期のプログラミングカリキュラムを図 5 に示す．

プログラミング教育体系とその狙い

最大の変更は，特別プログラムの開設である．アドバンスト教育として位置づけられ，成績上位者を 20 名程度選抜して実施することとなった．幸い，第 3 期まで安定的に 2～3 倍程度の応募者があり，手の込んだショッピングサイトを多数生み出すことができた．

プログラミング教育としては，多少の科目の移動はあっても，基本的に 2003～2007 年度カリキュラム(図 4)を引き継いだ形で，問題点も未解決のまま引き継いだ．

教育効果及び反省点

プログラミング以外にも「１」「２」を分ける科目が増加したが，単なるクラス分けと達成度別のものと 2 種類存在するようになった．

3.6　第 3 期 2012~2016 年度までのカリキュラム

特別プログラムに加え，実践ベンチャービジネスの導入，長期インターンシップの事実上の廃止となった．当

初に比べると，計算機科学の分野は科目としては縮小され，プログラミングと情報数学の学科のようになっている．この時期のカリキュラムを図６に示す．

プログラミング教育体系とその狙い

到達度別科目編成を，科目数的にも増やし，実践ベンチャービジネスの導入に伴い，科目を前倒しにした．これは，３年生になる前にプログラミング力をつけ，企業で実戦ベンチャービジネスを実施するに耐える実力をつける狙いがあった．

そのため，履修登録時の事前登録の機能を使って，「プログラミング入門」の成績による履修科目を履修登録時にあらかじめ登録されている状態にした．またテキストも定評のある市販のもの[2]を利用するように統一した．

教育効果及び反省点

「事前登録」の機能は魔法のように効き目を発揮し，「プログラミング入門」の成績上位者は必ず「プログラミング言語」を履修するようになり，「プログラミング言語」合格者は，「実用プログラミング１」はほとんど全員が履修，「実用プログラミング２」も離脱者は少数となった．成績上位者が上位科目を履修しない問題は，思わぬところで完全に解決した．

プログラム関係の科目が前倒しになったため，「アルゴリズムとデータ構造１」「同２」は，プログラミング的に履修内容が不十分な状態の学生に対して開講することになった．そこで，半期２科目 30 コマを前後２科目に分けて，実施した．かなりの詰め込みになり，講義する方も難しかったが，学習する方にも疲労感がかなり感じられた．

プログラムできるようになったか，については，教育の効果としては判断しにくい．できる学生ができるようになる当たり前の結果しか得られていない．どのような教育内容で，どの程度の効果が得られるものか，見極めるための一層の努力が必要な状態が続いている．

4. プログラミング教育の今後

前節で述べてきたように，第１期～第３期に渡るプログラミング教育の実践と改良を行ってきた結果，おおよそ次のような知見を持つに至った．

4.1 能力別編成の重要性

プログラミングの能力は個人差が大きい．能力が著しく欠如しているケースでは，当人の努力や教育によって補える範囲は限られている．現状では「プログラミング入門」の成績によって２クラスに分けられているが，これを３クラスに分け，「プログラミング入門」自体もクラス分けして習熟度別の実施体制を敷くべきである．Scratch[4]のような親しみやすい GUI 言語を使おうが，ゲーム構築のためのプログラミングであろうが，興味はあっても能力がなければ教育効果は期待できない．全体的に，達成感のある短いプログラムの構築を積み重ねて，地道に習熟させていくほかないようである．

4.2 プログラミング言語の選択

現在では C/C++といったコンパイラ言語よりも，Python や Julia といった動的言語の利用が全世界的に盛んである．MATLAB のように，数値シミュレーションに特化したソフトウェアも，動的言語の一種と言え，今後は II 類科目，理工学部の III 類科目での利用が増えていくと予想される．

しかしながら，プログラミング能力の涵養を狙うのであれば，機能が絞られたプリミティブな低級言語の基本文法から入門していくことが望ましいと考えている．従って，当分の間は，C/C++を中心にプログラミング入門は実施していく予定である．

4.3 「プログラミング入門」の選択化

どうしてもプログラミングができない，したくないという学生はかなりの割合で本学には存在する．従って，能力別編成と共に，Office の操作にもっと習熟できる科目を増やしたり，動的言語を操って短い出来合いのスクリプトを触る程度の科目を選択できるような履修制度を考えていく必要があろう．

5. 結語

現在の情報学部は，理工学部諸学科の倍以上の定員数を抱える割には，今一つ評価されていないように感じるのは著者らのヒガミなのかもしれないが，一番の特徴である「プログラミング教育」の成果を伝える努力を怠ってきたという事情も関係していると思われる．

本稿で縷々述べてきた情報系学科・学部における過去・現在・未来のプログラミング教育の議論が，高度情報化社会を生きる情報学部の学生への幾ばくかの手助けになれば幸いである．

参考文献

1) 静岡理工科大学　学生便覧，pp.180-181，2017
2) 柴田望洋，新・明解 C 言語　入門編，SB クリエイティブ，2014.
3) 菅沼義昇，「菅沼研究室へようこそ」，http://www.sist.ac.jp/~suganuma/
4) Scratch，https://scratch.mit.edu/