

基于计算机系统能力培养的程序语言选用分析

罗先录, 周富肯, 艾广燊, 向燕飞

(广东东软学院计算机学院, 广东 佛山 528225)

✉luoxianlu@nuit.edu.cn; zhoufukun@nuit.edu.cn;

aiguangyi@nuit.edu.cn; xiangyanfei@nuit.edu.cn



摘要: 计算机类专业的学生掌握编程解决问题的基本技能, 是基本的专业素质。算法是程序设计的灵魂, 是用系统方法描述解决问题的方法, 是从编程通向计算思维的必由之路。计算机类专业的“系统能力”, 是计算机类专业相较于其他专业的核心竞争力。本文总结了六年来采用不同编程语言进行教学的比较, 讨论了如何贯通专业学生的这些专业基本素质和核心能力, 为我们进行教学时编程语言选择提供了基于系统能力的可行方案。

关键词: 程序设计; 数据结构与算法; 计算机系统; 编程语言

中图分类号: TP312 **文献标识码:** A

Analysis of Program Language Selection from the Perspective of Computer System

LUO Xianlu, ZHOU Fukun, AI Guangyi, XIANG Yanfei

(School of Computing, Neusoft Institute Guangdong, Foshan 528225, China)

✉luoxianlu@nuit.edu.cn; zhoufukun@nuit.edu.cn;

aiguangyi@nuit.edu.cn; xiangyanfei@nuit.edu.cn

Abstract: It is a professional quality for learners of computer majors to master basic skills of problem solving with programming. Algorithm, the soul of program design, is a stepwise procedure for solving problems systematically, and is the only access to computational thinking. "System Ability" is the core competitiveness of computer majors compared with other majors. This paper summarizes and compares teaching with different programming languages over the past six years, and then discusses how to enhance both professional qualities and core abilities of computer major students. The result of the paper provides a feasible scheme for programming language selection of teaching for cultivation of students' system capability.

Keywords: program design; data structure; algorithm; computer systems; programming language

1 引言(Introduction)

数据结构与算法课程曾经用伪代码教学, 旨在掌握算法思维本身。现在基本采用一门具体的编程语言进行教学, 可能是Linux Torvalds说: "Talk is cheap, show me the code.", 也可能是程序执行时间比算法复杂度分析更有感觉。当前计算机系统能力的培养也是一个倒逼的动力, 因为体验代码的执行、调试程序的bug, 是让学生深入理解计算机系统, 加强工程能力培养的必由之路^[1-4]。

目前, 计算机类专业的程序设计基础和数据结构与算

法这两门课程的教学, 选择同一门编程语言进行两门课程的连续教学是较好的选择。这样为学生“学好一门编程语言”+“计算思维”这个主线提供了很好的条件^[5]。

那么是让学生自主选择课程的这一门编程语言, 还是根据专业课程体系的需要进行设定, 这是我们进行课程体系设计的时候容易引起争论的话题。我们根据TIOBE编程语言排行榜以及网上招聘和地区人才需求, “所学即所用”“一点打透”等原则, 在学生中试行了学生自由选择排名前列的几种编程语言进行这两门课程的教学。现在根据计算机

系统能力建设的趋势，回归了简单的C语言路线。这又会让人想起发明C语言的Dennis M. Richie说：“Keep it simple stupid”^[6]。我们做了一些思考和总结，供专家和同行们参考。

2 面向对象编程语言的特点(Characteristics of object oriented programming language)

我们以Java语言为例来说明用面向对象编程语言来进行程序设计基础+数据结构与算法课程教学的特点。Java的优势在于实现了面向对象的主要技术(封装、继承与多态、接口等)，而放弃了容易引起混乱的技术(例如多继承)。

(1)封装。在数据结构与算法课程教学中，数据抽象是首先要面对的主题。现在的面向对象技术已经很成熟，如果选择将数据成员(数据结构)设为私有属性隐藏起来，通过调用公有成员方法(算法)访问私有数据，Java完成了定义自己的数据类型(数据抽象)和对这些数据操作的封装。数据类型和相关操作(算法)有了密切相关性，定义了一个类的数据成员(数据结构)之后，相应的算法就是同一个类中的各种方法，就可以创建对象，以及派生出更多的细分、实用的类。

(2)泛型。泛型是Java 5以上版本提供的支持这一目标的方式之一。泛型也就是参数化类型，例如Stack<Item>是某种元素的栈，而Stack<String>处理的就是String对象。同理，Queue<Date>就是处理Date对象^[7]。

算法中经常需要进行对象的大小比较，一般采用两种方法实现泛型，从而实现对象间可比较。一是用Object表示泛型，二是用Comparable接口类型表示泛型。用Comparable接口类型表示泛型时(从JDK 1.5开始，在java.lang.Comparable中使用了泛型)，需要定义对象为Comparable接口的实例，然后调用Comparable的compareTo方法(需要重写)就可以进行对象间的比较操作^[8]。

基本数据类型(int, char, double, float等)可以使用比较运算符，基本类型的包装类(Integer, Character, Double, Float等)和String类都已经实现了Comparable的类，也实现了compareTo方法，可以直接使用。如果是自定义的类，就需要重写compareTo方法。

(3)接口与继承。虽然Java没有多继承，但是接口可以实现类间多重继承的功能，也就是通过指定一个含有一组公共方法的接口为两个本来没有关系的类建立一种联系。通过抽象数据类型，定义隐藏数据的存储结构并且不涉及操作实现细节的数据类型。例如定义线性表时，首先定义抽象数据接口^[8]：

```
public interface IList{
    ...
    public void insert(int i, Object x) throws
    Exception;;
```

```
public void remove(Object x) throws Exception;;
...
}
```

当采用顺序存储实现时，用SqList类来实现上述接口^[8]：

```
public class SqList implements IList{
    private Object[] listElem;
    // 线性表特有的私有数据，存储空间
    private int curLen;
    //线性表的当前长度
    //线性表的构造函数
    public SqList (int maxSize) {... ...}
    // 插入操作的函数
    public void insert (int i, Object x) throws Exception
    {.....}
    // 删除操作的函数
    public void remove (int i) throws Exception {.....}
    ... ...
}
```

当采用链式存储实现时，用LinkedList类来实现上述接口^[8]：

```
public class LinkedList implements IList {
    public Node head;
    // 单链表线性表特有的私有数据，头指针
    // 单链表的构造函数
    public LinkedList( ) {... ...}
    // 插入操作的函数,这些一致的函数接口，使得调用时
    没有差别
    public void insert (int i, Object x) throws Exception
    {.....}
    // 删除操作的函数,这些一致的函数接口，使得调用时
    没有差别
    public void remove (int i) throws Exception {.....}
    ... ...
}
```

上述抽象数据类型不同的实现中接口一致，这样使得学生在掌握了这个框架的不同细节之后，再去比较不同实现的各个算法(方法)性能差异和应用场景就有了一个全景图。

如果将算法应用于实际的场景，而需要扩展算法，Java的面向对象的子类继承特性就会发挥出来了，其中的每个方法需要根据具体的需求进行重载或者覆盖。例如，学生成绩查询系统继承于顺序表，包含了查找、删除、插入，以及创建学生管理系统，输出系统中学生信息的功能。其中的get、remove方法覆盖了父类SqList原来的方法，而get、display重载了父类SqList原来的方法。这样不同的应用场景只要继承扩

展原来的算法代码, 不需要改写原来的代码, 体现了面向对象技术的优点。如下代码所示^[8]:

```
public class StudentManagSystem extends SqList {
    // 按顺序构造顺序表, 其中参数maxSize指的是顺序表
    的最大存储空间容量

    public StudentManagSystem(int maxSize, int n)
    throws Exception { }

    //覆盖父类的get方法, 从顺序表中取出指定学号的学生
    信息, 并返回
    //一个StudentNode对象

    public StudentNode get(int number) throws
    Exception { }

    // 重载了父类insert方法在顺序表的表尾插入一个学生
    信息

    public void insert(StudentNode node) throws
    Exception {
        super.insert(this.length(), node);
        // 调用父类的insert方法
    }

    // 覆盖父类的remove方法

    public void remove(int number) throws Exception { }
    // 重载父类display()方法, 输出顺序表中的学生信息

    public void display() { }
}
```

(4)算法测试。Java的算法完成后, 测试或者调试算法很方便。只要在类中增加一个静态main方法, 就可以调用本类中实现的方法(算法), 一般也在这里进行单元测试。运行或者调试本类的编译后的class文件, 就可以测试算法的正确性和性能。

C#版本的优势和问题与Java比较类似, 我们在数字媒体技术专业中采用了C++语言, 通过模板和继承来实现数据的抽象, 特别是标准模板库(Standard Template Library, STL)可以实现数据抽象和通用性。比较其他面向对象语言, C++有一些实现技术细节更难以理解和更容易引起混淆, 例如多继承等。

总之, 面向对象技术可以使得数据结构与算法有机结合, 算法是对应数据结构的算法, 缺点就是学生要熟悉面向对象技术的同时去掌握不是那么容易理解的数据结构所对应的算法, 即使是已经采用同一种面向对象编程语言学过了程序设计基础, 但是要理解数据结构所需要的面向对象技术的高级部分, 也不是那么容易, 一个用C语言可以简洁表达的算法用C++、Java或者C#实现后, 看起来像一个英语文档, 很多学生缺乏同时学习两种新思维的能力, 结果算法的重点不

是算法思维本身, 而变成了承载算法的编程语言语法。

3 系统能力角度的C语言(C language from the perspective of system capability)

3.1 C语言存在一些面向对象技术所没有的问题。

(1)C语言中数据结构和算法没有封装在一起。C语言通常用结构体来自定义数据类型, 算法和数据结构之间没有封装; 实现同一种数据结构(抽象数据类型)的算法在不同的具体实现时, 实际上是完全独立的两套定义, 不是面向对象语言那样可以先定义一个抽象接口, 然后具体实现, 从而具有—种整体体系的感觉。例如用顺序和链式实现栈(Stack)的时候, 如果为了实现接口一致, 只能分开在独立的文件中定义两种不同的实现:

以下是顺序实现的栈, 假设在sequenceStack.h中定义^[9]:

```
typedef struct SNode *Stack;
struct SNode {
    ElementType *Data; /* 存储元素的数组 */
    int Top; /* 栈顶指针 */
    int MaxSize; /* 堆栈最大容量 */
};

Stack CreateStack(int MaxSize);
bool Push(Stack S, ElementType X);
bool IsFull(Stack S);
bool IsEmpty(Stack S);
ElementType Pop(Stack S);
```

以下是链式实现的栈, 假设在linkStack.h中定义^[9]:

```
typedef struct SNode * Stack;
struct SNode {
    ElementType Data;
    Stack Next;
};
```

其余函数(操作或者算法)的接口一致, 具体实现用不同的数据结构^[9]:

```
Stack CreateStack(int MaxSize);
bool Push(Stack S, ElementType X);
bool IsFull(Stack S);
bool IsEmpty(Stack S);
ElementType Pop(Stack S);
```

(2)C语言实现数据抽象或者通用的函数比较麻烦。—般把数据结构的定义和相关的算法写到一个或者多个头文件中, 解决问题或者测试算法的主函数写到另外一个C文件中, 例如上述同一个抽象数据类型的两种不同具体实现的头文件, 使用的时候需要用文件包含#include “sequenceStack.h”或者“linkStack.h”宏指令, 包含到主函数所在的C文件中

去, 由于两个定义中的函数接口一样, 所以每次只能包含一个, 那么只能修改 `#include` 语句或者用条件编译宏定义解决。如果解决问题的需要扩展数据结构或者算法, 只能修改原来的头文件或者也通过宏定义的方式进行预编译处理的方式进行。在方法之间进行数据传递的时候, 如果数据类型不确定, 就一般用 `void*` 指针传递后再进行强制类型转换。

(3) C语言也没有Java和C++那么丰富的库函数功能。自带的库函数没有实现多少算法, 即使是常用的 `qsort` 库函数, 也需要自定义比较函数指针所指向的函数(第四个参数)。这个特点本质上和上面第二点是一样的。以下代码来自浙江大学数据结构课程(中国大学MOOC, 陈越)^[9]:

```
int compare(const void *a, const void *b)
{ /* 比较两整数。非降序排列 */
    return (*(int*)a - *(int*)b);
}
```

调用库函数时, `compare` 函数指针指向上面实现函数:

```
qsort(A, N, sizeof(int), compare);
```

如果是别的数据类型或者复杂的数据, 例如结构体中多个键值, 需要重新定义上述实现比较的函数。

3.2 C语言的优点

1973年, Dennis M. Ritchie和Ken Thompson为了改写Unix系统而发明了C语言, 他们将C语言和Unix完美的结合在一起, 产生了可移植的Unix系统, 从而改写了计算机的历史。Keep it simple stupid原则也是Dennis提出的, 所以C语言具有简单易学, 和操作系统完美结合等优点。

(1) 效率。由于C语言为了操作系统而生, 所以更接近硬件, 能够直接访问内存和进行内存地址运算。C语言和汇编语言关系密切, 可以在C语言代码中嵌入汇编代码。C语言还支持比特(bit)级的移位操作等, 所以和其他面向对象的语言比较, C语言很高效。

(2) C语言编译过程中的汇编语言代码、机器代码和C语言代码对应很完美。学生在学习了C语言和编译后的汇编对应的模式之后, 就容易理解编译过程、逆向工程。我们看到在计算机类专业课程体系中, 程序设计基础和数据结构与算法课程后续的专业核心课程, 例如组成原理、操作系统、计算机网络、编译原理、体系结构等, 一般采用C语言举例或者阐明相关原理。学习这些课程, C语言作为一根主线, 更能够深刻理解计算机系统和整个专业的课程体系, 为后面的专业方向课程甚至是工作后的解决问题能力打下坚实的基础。

3.3 计算机系统角度的C语言

从程序员角度的计算机系统看, C语言版本的优势还在于可以深刻理解计算机系统, 这个是当前计算机教育的一个趋势^[1-3]。2010年, 教育部高等学校计算机类专业教学指导会组织高校的资深计算机教育专家成立“计算机类专业系统能力

培养研究专家组”, 研究组经过对国内外计算机专业的课程体系、用人单位的需求及技术的发展, 提出了“计算机类专业系统能力培养专业课程体系”。并分批设立了试点高校。如果是计算机系统能力试点学校, 就会了解目前所有的计算机系统基础之类的课程和教材, 都是采用了C语言举例^[1,2], 特别是其中最著名的CMU教材《深入理解计算机系统》就是采用了C语言作为例子贯穿整个教学^[10]。我们以南京大学袁春风《计算机系统基础》为主教材, 同时参考CMU的《深入理解计算机系统》, 加入了第四批试点高校行列。我们结合这几年进行程序设计基础+数据结构, 以及后续的计算机系统基础课程这三门课连续教学中的一些主要概念的逐级提升或者延伸、最后得以澄清的经验体会来说明这个选择的优点。

表1 贯穿专业教学的一些概念背后的思路和机制

Tab.1 Ideas and mechanisms behind some concepts of professional teaching

概念	层次	C程序设计基础涉及内容	数据结构与算法涉及内容	计算机系统基础涉及内容	串起内容的线索
线性结构与链式结构的访问		数组和指针(结构体、链表)	线性表、栈、队列、树和图等抽象数据类型等都有两种结构的实现方式	顺序实现的数组首地址以及循环模式和链式实现的链式访问模式的比较	在汇编代码级理解各种数据访问模式的本质
栈(stack)		在调试运行递归函数时观察调用的函数栈	LIFO问题的解决模式, 熟悉栈的底层实现和运用	操作系统的栈是一个特定内存区域, 和数据结构的栈一样遵循先进后出的规则	数据结构的栈和操作系统函数调用栈的区别、函数调用栈的耗时和溢出是线索
堆(heap)		动态内存分配函数 <code>malloc</code> / <code>free</code> (面向对象编程语言的 <code>new</code> 的使用)	一种特殊的二叉树数据结构, 以数组方式连续存储	操作系统的堆是一个特定的动态内存分配区域(运行时才知道大小需求)	操作系统的堆(栈)强调数据生命周期, 而数据结构的堆(栈)强调数据组织
排序算法中的归并排序		多项式加法等简单运用	递归实现或者非递归实现	额外空间的分配, 需要理解内存的分配和回收的代价	从系统角度理解“以空间换时间”和“以时间换空间”的权衡
堆排序和快速排序比较		两个变量比较和交换的方法	对于同样的数据, 排序过程中, 堆排序算法的交换次数要多于快速排序	快速排序中数据是局部顺序访问, 堆排序是跳跃访问, 对CPU缓存不友好	从系统角度理解算法快慢的根本原因
文件		文件建立/打开/关闭、读/写、文件指针移动	内部排序和外部排序	Cache、内存、I/O的读写	Cache/内存/硬盘等存储层次结构的效率和配合

这样的例子还可以不断总结很多, 运用C语言实现的例子, 就很容易阐述清楚这些概念背后的内涵与延伸, 让学生深入理解计算机系统的运行机制, 逐步形成计算机专业的核心能力。

4 结论(Conclusion)

对于一般专业学生来说,面向解决问题的模式,学会一种编程语言后进一步学习算法和数据结构,用一种语言学习这两门课程是一个很顺利的学习进阶道路,目前的教材和师资的成熟度也支持这样的做法。对于计算机类专业学生来说,用C语言进行这两门课程的学习,对于后续专业核心课程的学习,将会起到很好地理解与掌握计算机系统的作用。特别是目前的计算机教育更强调计算机系统能力的培养的形势下,更是一个必然的选择。

参考文献(References)

- [1] 王志英,周兴社,袁春风,等.计算机专业学生系统能力培养和系统课程体系设置研究[J].计算机教育,2013(09):1-6.
- [2] 臧斌宇,彭远红.反思、借鉴、融合、提升—探寻计算机系统能力培养之道[J].计算机教育,2015(21):1-2.
- [3] 袁春风,王帅.大学计算机专业教育应重视“系统观”培养[J].中国大学教学,2013(12):41-46.
- [4] 尚凤军.面向计算机系统能力培养的课程和实践体系研究[C]. Proceedings of the 3rd International Conference on Applied Social Science Research(ICASSR 2015), 2015:4-5.
- [5] 汤伟.《数据结构》和《C语言程序设计》新教学模式研究[J].科技资讯,2017,15(24):170-171.

(上接第12页)

也必将提高,气体传感器将会提供从空气污染到室内环境,从食品安全到人体健康等方面准确可靠的信息,将为物联网带来再一次的大发展。

5 结论(Conclusion)

物联网是信息获取、信息传输和信息处理的高速信息闭环系统。互联网本身就是高速的信息传输通道,计算机的处理速度一直是其发展的主线。随着两者的发展,注定了高速信息获取技术的突破,以实现整个系统的速度匹配性。

传感器是物联网的信息获取主要手段,而气体传感器能够对气体的检测便捷性和无损性,满足了气体及气体间接相关事物的信息快速获取需求。气体传感器将为物联网带来更加多元化,深层次的信息,也必将为物联网的发展提供非常重要的发展助力。

参考文献(References)

- [1] 王兆义.物联网及其应用[J].电世界,2011,052(008):11-14.
- [2] 章天金,周东祥.金属氧化物气敏传感器的研究现状及其发展趋势[J].材料导报,1998,012(006):38-40;44.
- [3] 赵志军,沈强,唐晖,等.物联网架构和智能信息处理理论与关键技术[J].计算机科学,2011,38(8):1-8.
- [4] 许磊.论物联网的关键技术及计算机物联网的应用[J].数码世界,2020(1):5.

- [6] Brian W. kernighan, Dennis M. Ritchie. The C Programming Language(Second Edition)[M]. New Jersey:Pearson Education International,1998:23-189.
- [7] Robert Sedgewick, Kevin Wayne. 谢路云,译.算法(第4版)[M]. 北京:人民邮电出版社,2014:38-107.
- [8] 刘小晶,杜选,朱蓉,等.数据结构—Java语言描述[M].北京:清华大学出版社,2015:23-64.
- [9] 陈越,何钦铭,徐镜春,等.数据结构(第二版)[M].北京:高等教育出版社,2019:73-276.
- [10] Randal E. Bryant, David R. O'Hallaron. 龚奕利,贺莲,译.深入理解计算机系统[M].北京:机械工业出版社,2018:99-276.

作者简介:

罗先录(1973—),男,硕士,教授.研究领域:网络空间安全,人工智能.

周富肯(1980—),男,硕士,副教授.研究领域:计算机应用与技术.

艾广燧(1984—),男,博士,讲师.研究领域:嵌入式系统,数字信号处理,生物信息处理.

向燕飞(1980—),女,硕士,讲师.研究领域:计算机软件与理论.

[5] 郑广成,朱翠苗.基于物联网结构的智慧教室建构的研究[J].软件工程,2016(19):44-46.

[6] 陈春云.物联网在智能家居中的应用与发展探析[J].信息记录材料,2019,020(002):73-74.

[7] 苏春芳.基于多传感器融合的居家监护系统的设计与实现[J].软件工程,2019(11):18-21.

[8] 马须敬,徐磊.气体传感器的研究现状与发展趋势[J].传感器与微系统,2018,37(315):8-11;19.

[9] 马泽亮,国婷婷,殷廷家,等.基于电子鼻系统的白酒掺假检测方法[J].食品与发酵工业,2019,45(02):194-199.

[10] 王佳音,王凤玲,王福东,等.电子鼻对不同种类抹茶香气成分的辨别与分析[J].食品与发酵工业,2019(19):1-10.

[11] 吕斌,孙玉国.室内有害气体云端监测系统设计[J].软件,2019(12):158-162.

[12] Jia W, Liang G, Jiang Z, et al. Advances in Electronic Nose Development for Application to Agricultural Products[J]. Food Analytical Methods, 2019,12(6):2226-2240.

作者简介:

赵林(1978—),男,博士,副教授.研究领域:多传感器信息处理,物联网.