第 1、3 次课

1、What is Computational Thinking?

Computational Thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.

Informally, computational thinking describes the mental activity in formulating a problem to admit a computational solution. The solution can be carried out by a human or machine, or more generally, by combinations of humans and machine.

2、Computational thinking for everyone means being able to

（1）Understand what aspects of a problem are amenable to computation
（2）Evaluate the match between computational tools and techniques and a problem
（3）Understand the limitations and power of computational tools and techniques
（4）Apply or adapt a computational tool or technique to a new use
（5）Recognize an opportunity to use computation in a new way
（6）Apply computational strategies such divide and conquer in any domain

3、Computational thinking for scientists, engineers, and other professionals further means being able to

（1）Apply new computational methods to their problems
（2）Reformulate problems to be amenable to computational strategies
（3）Discover new “science” through analysis of large data
（4）Ask new questions that were not thought of or dared to ask because of scale, easily addressed computationally
（5）Explain problems and solutions in computational terms

4、人如何解题（George Polyad:" How to solve it?"）（去年计算思维考题）

（1）Understanding the problem(理解问题):"What you are given（给出的条件） and what you are supposed to figure out（想要得到的结果）"
（2）Devising a plan(拟定计划):" How will you attack the problem？（怎样面对问题）"
（3）Carrying out the plan(执行计划):" Solve the problem（解决问题）">>>计算机只能解决这个问题，其他问题需要人来解决
（4）Looking back(回顾与校验):" Check the result, and ...（检查结果，反思问题等等）"

5、计算机如何解题？

（1）理解问题：输入是什么？处理规律是什么？输出是什么？
（2）制定计划：设计算法，设计程序，设计数据库，设计接口等等
（3）执行计划：计算机解题
（4）检查结果：check out

6、数据抽象的核心概念：（通过解释进行数据抽象）

信息形态、信息组织、存储、检索与利用

7、算法是计算思维的核心概念，是解读计算思维的最佳载体（通过算法进行问题抽象）

8、系统抽象的核心概念？（通过平台进行系统抽象）

系统模型、功能逻辑、接口、实现

9、计算思维是我们认知计算过程中的积累的思考“模式”，计算思维=抽象化+自动化

10、计算一个图片的大小？

e.g：532x52824 位图=532*528*24/8(8bit=1Byte)/1024(1KB=1024Byte)=822KB

11、利用 flip, zero, test，exit 和 goto 五个指令完成二进制整数的加法（今年作业）、减法、比较（去年的计算思维考题和今年作业）等功能。

12、完成二进制数和十进制数（包含小数的情况）的互相转化（去年计算思维考题）

第 2 次课

1、2012-2017 IT 趋势

（1）2012：大数据
（2）2013：移动设备
（3）2014：智能机器
（4）2015：风险保护
（5）2016：机器学习
（6）2017：人工智能

2、现象、数据、信息、知识、智慧

（1）现象=表象+感知
（2）信息=数据+处理
（3）知识=信息+理解+推理+解决问题的技能

第 6 次课

1、程序=算法+数据结构

2、算法的部分正确性和完全正确性：

（1）部分正确性：if this point is reached, then the current output is the desired output
（2）完全正确性：indeed this point is reached, and the current output is the desired output

3、如何判断算法的正确性？（此处暂时指利用循环不变式进行证明）

（1）依赖于算法 check points 的 assertion（断言）的序列

a）Check points 应该设置在初始化、循环的每个节点、循环后结果三个地方推理的过程

b）类似数学归纳法的证明过程，但多了终止时的正确性判断的步骤

4、循环不变式：必须证明以下三条性质：

（1）初始化：循环的第一次迭代之前，它为真

（2）保持：如果循环的某次迭代之前它为真，那么下一次迭代之前它仍为真

（3）终止：在循环终止时，不变式为我们提供一个有用的性质，该性质有助于证明算法是正确的

5、算法框图的绘制——自行复习

6、掌握用伪代码或 C 语言写程序的能力——去年的计算思维的解题要求

7、INSERTION-SORT（非降序排序）的伪代码描述？

```
INSERTION-SORT(A)
    for j = 2 to A.length
        key = A[j]
    //Insert A[j] into the sorted sequence A[1..j-1]
    i = j - 1
    while i > 0 and A[i] > key
    A[i+1] = A[i]
    i = i - 1
    A[i+1] = key
```

8、考虑把两个 n 位二进制整数加起来，这两个整数分别存储在两个 n 元数组 A 和 B 中，这两个整数的和应按二进制形式存储在一个 n+1 元数组中。请给出该问题的伪代码？（《算法导论》）

```
ADD-BINARY(A, B):
    C = new integer[A.length + 1]
    carry = 0
    for i = 1 to A.length
        C[i] = (A[i] + B[i] + carry) % 2  // remainder
        carry = (A[i] + B[i] + carry) / 2 // quotient
        C[i] = carry
    return C
```

9、循环不变式的作用？

帮助证明算法的部分正确性

10、算法的必要考虑因素？

（1）算法的优化设计
a）算法本身的优化设计
（2）数据组织导致的优化

11、用哪些数值标定算法的性能？

（1）评价算法分析性能的标准主要从算法执行时间和占用存储空间两个方面进行考虑，即通过时间复杂度和空间复杂度来判断一个算法的优劣。（百度百科）
（2）可以通过算法的评估函数渐进增长速度的比较来比较算法的优劣（ppt 内容）

12、什么是算法的渐近时间复杂度？

当输入规模足够大，使得只有运行时间和增长量级有关时算法的渐近效率。也就是输入规模无限增加时，在极限中，算法的运行时间如何随着输入规模的变大而增加。（《算法导论》）

13、简要写出你计算一个算法的渐近时间复杂度的过程？（去年试题）算法的 best case 和 worst case 如何计算？

14、函数的渐近增长速度？

给定 $f:N\rightarrow R+$, $g:N\rightarrow R+$,如果存在常数 $c \in R+$ 和 $k \in N$ 使得对于所有大于等于 k 的 n，都有 $f(n) \leq C \times g(n)$，我们称:f is O(g)；f 渐近增长速度不高于 g

15、数据的组织和结构涉及的内容？

（1）数据自身的逻辑模型
a）什么样的顺序、排列方式
（2）数据在计算机中的存储模型
a）数组方式
b）栈的方式
c）先进先出表的方式

16、归并排序

（1）归并排序的原理？甚至用伪代码书写 MERGE 过程？

```
MERGE-SORT(A, p, r)
1  if p < r
2       q = ⌊(p + r)/2⌋
3       MERGE-SORT(A, p, q)
4       MERGE-SORT(A, q + 1, r)
5       MERGE(A, p, q, r)
```

```
MERGE(A, p, q, r)
 1  n1 ← q − p + 1
 2  n2 ← r − q
 3  create arrays L[1.. n1 + 1] and R[1.. n2 + 1]
 4  for i ← 1 to n1
 5       do L[i] ← A[p + i − 1]
 6  for j ← 1 to n2
 7       do R[j] ← A[q + j]
 8  L[n1 + 1] ← ∞
 9  R[n2 + 1] ← ∞
10  i ← 1
11  j ← 1
12  for k ← p to r
13       do if L[i] ≤ R[j]
14            then A[k] ← L[i]
15                 i ← i + 1
16            else A[k] ← R[j]
17                 j ← j + 1
```


（2）如何考虑分治算法的代价？

递归代价与非递归代价

（3）分析分治算法的分析？

a）分解

b）解决

c）合并

（4）对于归并排序，有此递归式：

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T(n/2) + \Theta(n) & \text{if } n > 1 \end{cases}$$

17、什么是 f 和 g 同阶？

let f(n)=n2, g(n)=7n2 +9n-1

lim(n→∞)[f(n)/g(n)]= lim(n→∞)[n2/(7n2 +9n-1)]= 1/7，所以： f 是 O(g)

lim(n→∞)[g(n)/f(n)]= lim(n→∞)[(7n2 +9n-1)/n2]= 7，所以： g 是 O(f)

由此我们称：f 和 g 长得一样快(同阶)

18、什么是 algorithmic gap？

问题 P 的难度是固有的，假设是 g；这个 g 是我们竭力想知道，但往往我们却难以认识到。当我们找到一个解这个问题的算法时（假设它的性能是 f），我们可以说的是：g 属于 order of f。f 是算法问题难度的上界；当我们不断优化算法得到 f',f''，……时，算法问题的难度上界不断降低（order 级别的降低）但是，仅仅降低上界，不能得到 g。同时，如果我们能够证明，解这个算法问题，至少需要 order of h 的算法才可以，比如遍历的难度至少是 O(n)，那么我们就得到了算法问题的一个难度的下界。那么我们在这个方向上的工作就是不断去证明：问题 P 的解的最小代价是否比 h 高，得到下界 h',h''。如此，问题 P 的解的下界不断在提高（用证明来完成），问题 P 的解的上界不断在下降（用算法设计来完成），如果他们 meet 了，我们可以说：我们找到了这个问题 P 的解的难度 g；否则，h'''和 f'''之间的 gap 就存在，就被称为这个问题的算法 gap。当算法 gap 存在时，要么就是我们没有找到最好的算法，要么就是我们不能证明还有其它更好的算法（无法证明现在的算法不能再好了），要么就是两者都没有完成。

19、算法的效率和效率的优化是算法设计不懈追求

20、几个优化算法的原因和做法？

（1）全局优化
- a）优化数据结构
- b）优化算法设计
- c）优化选择结构
- d）优化循环结构

（2）局部优化
- a）选择尽量小的数据类型
- b）使用自增、自减和复合赋值运算符、运算量较小的表达式
- c）避免浮点运算
- d）优化赋值语句
- e）优化函数参数

第 7 次课

1、逻辑清晰和结构清楚是算法最重要的内容

2、子程序的优点：复用、封装、抽象、丰富了算法的结构：顺序、选择、循环、调用

第 8 次课

1、Industrial-strength software 强在哪里？软件质量

2、软件质量如何去解读和度量？

（1）Functionality 提供满足设计目标的功能的能力
（2）Reliability(可靠性)：Failure free 的能力
（3）Usability(可用性)：被理解、学习和使用的能力
（4）Efficiency(高效性)：相对于消耗的资源，提供相应的性能的能力
（5）Maintainability(可维护性)：因为纠错、优化和适应等原因需要修改时，易于修改的能力
（6）Portability(可移植性)；适应不同平台和环境的能力
→其中，软件可靠性成为最关键的质量指标

3、如何评估软件的可靠性？

erally accepted to be the main quality criterion. As unreliability of software is due to the presence of defects in the software, one measure of quality is the number of defects in the delivered software per unit size (generally taken to be thousands of lines of code, or KLOC). With this as the major quality criterion, the quality objective is to reduce the number of defects per KLOC as much as possible. Current best practices in software engineering have been able to

4、软件产品不会产生损耗，为什么也需要维护？

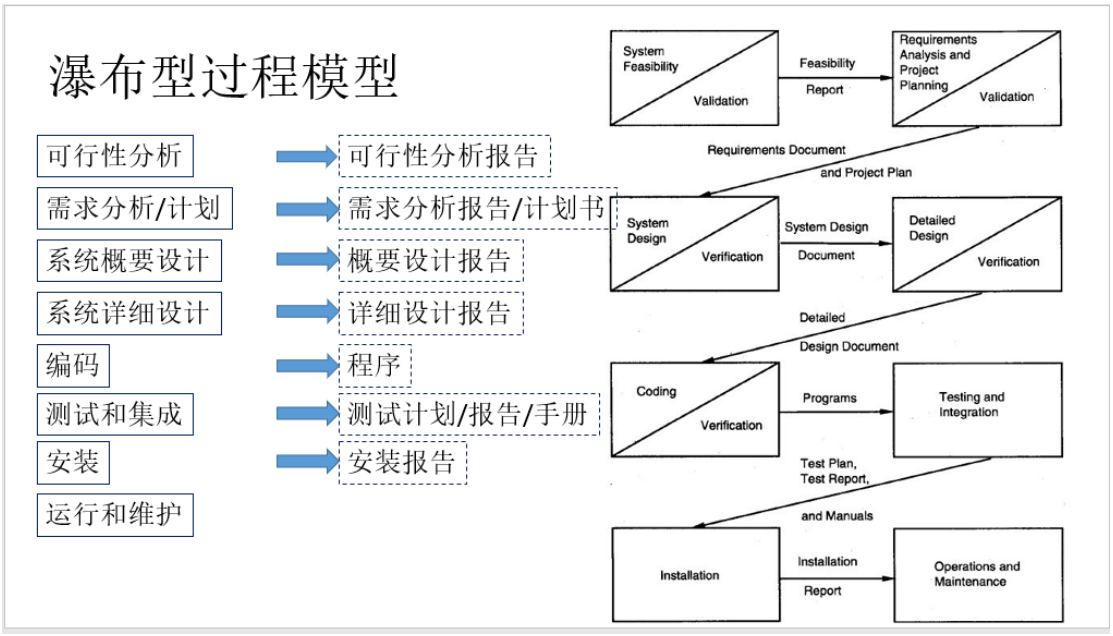
（1）正确性维护：修正错误
（2）适应性维护：满足需求和环境的变化
（3）软件维护：软件提交之后，为修正错误、满足新需求和适应新环境而进行的更新

5、软件产品生产的三要素：开销、进度、质量

6、为什么软件开发必须被“过程化”？

将软件开发分解为若干步骤，每个步骤赋予其特定的目标，界定每个步骤在工期、开销和质量三个方面的要求，进行严格的检查和管理，只有这样才能够成功达到预期目标地开发一个软件。

7、软件开发的模型？



第 9 次课

1、计算模型：图灵机模型

存在一个通用的计算机，针对任何图灵可计算函数，我们总是可以找到一个程序，在这个计算机上对这个函数进行计算求值。

这张纸带只是需要逻辑上在同一个空间即可，只要读写头能找到它即可，物理上没有限制

2、我们怎样才能做到（实现）这种“跨机器”计算？

必须准确地实现两个计算“分量”之间的控制和数据转移！即寻址和传递