感谢小松子、陆神、鼎爷（和他的基友）、文潇、龙哥这份文件结合了作业，syllabus和ppt，归纳了可能考到的重点，以及比较难以理解的概念。希望大家能够从中获益~
大家还是要自己花些时间看看书，如果没有时间就看看之前的作业和模拟题，遇到不懂的去查书或者傍大腿。 乔小奕

chap1
1、一定要记住与门、或门、非门和异或门分别是长成什么样子的!
2、flip-flop： 可以通过不断的实验，用不同的输入组合来探索一个flipflop的外部特性.

解释一下书上举的例子，先定义一下，上边为输入A，下边为输入B，输出为P
假设一开始A，B为0，则P为0.
（B保持零）置A为1，则P为1；重置A为0，则P仍为1.
此时，（A保持0）置B为1，则P变回0，重置B为0，则P仍为0.

大家可以再试试，可以发现任意情况下，让A“闪烁”一下，P都变成1且保持1；让B“闪烁”一下，P都变成0而保持0.（闪烁就是变成1再变成0）是不是很有意思~

以上是flipflop的set特性，下面是hold特性（这两个特性在ppt中有总结）

如果保持A为1， 不论B是什么输入 ， P恒为1。
如果保持B为1， 不论A是什么输入 ， P恒为0。
这也是很有意思的一个特性。

3、了解磁盘和cd的不同存储方式

磁盘以磁介质存储信息，以同心圆排列，每个同心圆称为track（道），因为外侧周长较长，于是信息较稀疏. 这是为了保持数据传输速率恒定（磁盘按匀角速度运转). 整个磁盘分为各个扇区（就像比萨），于是每条道上的扇区个数是相同的，但每个扇区上的信息量是不同的.

cd以光学介质存储信息，螺旋状，信息存储的线密度是相同的，于是cd外侧存储更多信息。
//于是要求根据激光束的位置改变转速（ppt上是这样说的，但书上说大部分cd驱动器是匀角速度，但它们适应了传输速度的波动，所以不会出问题）

4、整数存储

补码和移码是一定要掌握的重点！建议翻出作业来看!
//可以观察一下在补码中，相反数相加会overflow！但是如果舍去overflow的那位就变成0了。

5、小数存储

浮点数的存储是一定要掌握的重点！注意exponent field用的是移码。建议翻出作业来看.

6、数据压缩

数据压缩分为无损和有损。文档类的文件一定要用无损压缩，否则会被破坏。音频、视频类的文件经过有损压缩对影响效果的影响不大。
run-length encoding（行程长度编码），把连续相同的数据替换成一段代码，这段会代码指出重复的内容，和重复的次数.
Huffman code（赫夫曼编码）根据内容出现的频度来编码，出现的越频繁，编码越短.

chap2 数据操作

1、冯诺依曼结构的三个特性

（1）four subsystems（四大结构）：memory，ALU，CU，I/O System
//如果问五大结构：那么就把输入和输出算作两个。
（2）stored program（存储程序）
（3）sequential implementation（顺序执行）

2、V8机器码

和做图灵机的题目类似，认真按照题目和规则，模拟V8的运行，13条指令一般会给出的。可以重做一遍作业题练练手.
（1）注意Load指令有两种，后两位的16进制数有两种解读方式：一种把它解释为地址，读入该地址的数据；另一把它解读为数据，直接读入了.
（2）理解一下jump语句：看某一寄存器内的值是否与0号寄存器相同，相同就跳转到某一指令（jump语句里会给出）. jump语句配合add语句可以实现循环，这在作业题里是有的.

chap3 操作系统

1、操作系统的发展

job(作业）每次一个人进屋（当时的计算机很大。。），做所有的事情，包括调试好设备，输入穿孔卡片和磁带等。其他人排队。。

batch processing（批处理）派一个计算机操作员专门负责操作电脑（此时已经出现了操作系统）用户提交程序、数据和说明文件给他就可以了，不用亲自进屋操作。

interactive processing（交互式处理）出现了新的可以和用户进行互动的os，在程序执行期间也可以和用户发生交互，比如要求用户输入数据、指令等，不想之前那样，提交了程序只能得到确定的结果

realtime processing（实时处理）上述的os性能足够好的时候，就能达到实时处理

multiprogramming（多道程序设计）其实就是引入了timeslice，之后会讨论。

2、操作系统的组件
（1）软件的分类：P114 fig3.3
注意区别 application和utility software的区别，utility是对操作系统的拓展. 如果想加深理解的话可以看书上的例子P115. 刻录软件、解压缩软件、网络连接软件属于utility（还是去看看书吧。。）
（2）操作系统的组件:操作系统分为shell和kernel

shell（外壳），即命令解释程序，功能是把用户的行为解释为指令。
有基于文本的，比如MSDOS，和借助GUI（graphic user interface）的，GUI有个很重要的组成是window manager（窗口管理程序），就是我们电脑上的那些窗口。。

kernel（内核）

file manager(文件管理程序)
协调计算机和mass storage的使用。file manager管理了所有文件的信息，包括位置、权限等，虽然这些信息存在mass storage上，但file manager可以检索这些信息，然后全面了解这个mass storage（比如一个移动硬盘）. 其他软件要想访问文件，都必须经过file manager，需要经过它的批准，并且从file manager那里知道怎么样找到文件和如何操纵它。

device manager（设备驱动系统）
负责与外围设备和它们的控制器来通信，比如打印机，显示屏等

memory manager (内存管理系统）
管理内存，特别是分时处理的时候。分时处理时，有多个进程都要被处理，它们都保留在main memory上，memory manager负责给它们分配空间，记录它们的位置，并保证它们不会越出分配的空间。

调度程序和分派程序
之后的分时处理会仔细讨论的。

3、bootstrap过程

在main memory中有一块不能被更改的区域ROM，里面存着bootstrap程序，开机时，先执行Bootstrap程序，然后由它把os从mass storage整个读进main memory，之后它把系统的控制权交给os.

重要：请区分register（寄存器），main memory（主存储器），和mass storage(海量存储)
register：在cpu里，稍稍熟悉V8机器码，就会知道cpu只能对register的数进行运算. V8中，register从0到F.
main memory：就是我们常说的“内存”，现在的电脑一般都是4G或8G。main memory中有很多很多的cell，里面有很多很多的flipflop. 你给出地址，马上就能找到那个cell并读出数据。V8中，main momory 从00到FF，很多指令需要从main memory读到register中.
mass storage：硬盘，既有就是在机箱里的硬盘，也有CD，U盘等外部存储器. 一般信息都是以磁学或光学信息存在上面的，访问起来比从main memory上要慢.

4、协调进程（分时处理）

程序是一系列指令的集合，进程是执行程序的行为，是随时间的推进而动态改变的. 对于某一个进程，在每一个时间点都有一个进程状态，这个进程状态包括程序的当前位置和各种存储单元的值等.

multiprogramming

cpu的运算以时间片（time slice）为单位进行，每个timeslice只进行一个进程，当前timeslice结束后，cpu执行另一个进程.

cpu的两个秘书
(1)dispatcher（分派程序）
告诉cpu该做什么，比如什么时候中止当前timeslice，下一个timeslice该做些什么

(2)scheduler（调度程序）
维护着一个进程表，里面是需要执行的进程。刚才说的dispatcher在每个timeslice结束后，会从这个表里面选择优先级最高让cpu去做. 进程表里面的进程有两种状态：ready（就绪）和waiting（等待). 如果在ready状态，就根据优先级排好队，依次进入cpu；如果在waiting状态，比如一个进程想要从键盘输入一个数，那么等到waiting结束后就会转为ready状态（比如得到了键盘输入）

如果感觉上面的dispatcher和schuduler比较难懂，那么我们来看一个过程。

1、某一个timeslice开始，cpu执行dispatcher分给它的进程，计时器开始计时（下一步有2a和2b两种可能）

2a 计时器计满了一个timeslice的时长，发出interrupt信号，cpu接收到了信号后，做一些收尾工作，比如保存一下现在执行到哪儿了，现在的变量都是什么值等等，接着把控制权交给Dispatcher.

2b 进行了一个i/o操作，自动终止timeslice，把控制权交给dispatcher.

3、dispatcher现在掌握了控制权！它访问schduler维护的那个进程表，看看那个进程最优先，交给cpu，回到1.

4、在3的同时schuduler也有所动作，它把cpu刚刚结束的那个进程的优先级降低，其他优先级升高.

5、在整个过程中，如果某一个处于waiting状态的i/o操作执行完了（比如得到了键盘输入），那么立即解除waiting转为ready并升为最高优先级！（请注意你之前把这个i/o操作的timeslice强行中止了哦，是不是要补偿一下它？）这样下一个timeslice就会执行这个io的后续操作了.

chapter4
1、ipv4和ipv6
ipv4共有2^32个，ipv6共有2^128个
格式分别形如：
192.255.1.1;（4段3位十进制数）
2402 : f000 : 1 : 4418 : 280 : 8eff : fe8a : 90e5（8段4位16进制数）

2、packets 可能通过不同的路径传给（routed）用户（知道一下= =）

chapter5 算法
5.1 算法的四个关键字：
ordered（有序的）：没什么好解释的
unambiguous（无歧义的）：每一个步骤被唯一地完整地描述.
//书上说，NP问题中的不确定算法是另外一个研究论题，这里不考虑.
executable （可执行的）：不能牵涉到无穷、不能从事非法的操作.
terminating （可中止的）：算法应当可以中止.

5.2 伪代码（pseudocode）

1、以procedure开头，后接过程名称，括号内列出参数
2、赋值语句使用箭头
3、所有的语句块都使用括号
4、（可选）if和while语句结束后加上end if 或 end while

5.4/5.5复习程设的时候一起复习了吧！

5.6/12.5（这两章关系比较紧密，syllabus里给放在一起了）

1、算法复杂度（complexity）：(这是很混乱的地方，请大家逐行阅读）
对于某一算法，能算出一个确定的复杂度，记为Θ(f(n))
对于某一类问题，存在一系列可行的算法，如果某一算法的复杂度为Θ(f(n))，则称问题属于O(f(n)),如果问题的最优解法复杂度为Θ(f(n))，则称问题属于Θ(f(n)).

比如，排序问题可以有多种算法，比如插入排序算法和归并排序算法.
就插入排序算法而言，它的复杂度为Θ(n^2).
就归并排序算法而言，它的复杂度为Θ(nlg(n)).
于是，对于排序问题，它既属于O(n^2)也属于O(nlg(n)).
但是归并算法被证明是排序问题的最优解，于是排序问题属于Θ(nlg(n))！
//Θ这个记号有两个意义，一个针对算法，一个针对问题。前者表示算法的平均复杂度，由算法本身决定；后者表示某一问题的各种算法中最优解的复杂度.
O这个记号只有一个意义，仅针对问题，只要它有某一个算法复杂度为Θ(f(n))，那么该问题属于O(f(n)).一个问题可以属于很多的O.
//有一些问题被证明有最优解，于是采用了Θ记号，有一些问题没有最优解.

2、问题的分类，强烈建议阅读fig11.12(p481)

问题分为可解的和不可解的.
可解问题分为多项式问题和非多项式问题，看它能不能在多项式时间内解决.
多项式问题简称为P问题.
但是存在一些特殊的非多项式问题，如果对它们使用某一系列不确定算法（比如引入随机或发挥创意），可以在多项式时间内解决.而对于所有的多项式问题，加入一些无关紧要的不确定算法，仍然保持它的性质，能够在多项式时间内解决. 于是那种特殊的非多项式问题，连同所有的多项式问题，组成一个集合，这个集合内的问题，尽管引入了不确定的算法，也能在多项式时间内解决. 这样的集合，成为非确定性多项式问题（nondeterministic polynomial problems)，简称NP问题.
//NP问题不等同于非多项式问题！虽然后者的英文名称为nonpolynomial ,这只是一个美好的巧合。
//根据以上讨论中的某句话，P问题属于NP问题

存在一些奇妙的NP问题，如果对这些问题，能够找到多项式时间内的解法，可以将这个解法推广到一切NP问题，则称这些NP问题为NP-complete问题，简称NPC问题.
//根据上述定义，如果解决了某一个NPC问题（找到它的一个多项式时间解）那么一切NP问题都可以找到一个多项式时间解！于是一切NP问题都是P问题，而P问题都是NP问题，于是P=NP！于是共产主义实现了。。

chap6

1、 第一代语言： 机器语言(machine language)；
第二代语言：汇编语言(assembly language)；
第三代语言：高级语言(advanced language)；

2、编程范式：强烈建议至少看一看5.1的课后习题和答案（p217）下面的翻译仅供参考
命令型范型（imperative paradigm）强调解决问题的一步步过程
说明型范型（declarative paradigm）强调对问题的描述
函数式范型（functional paradigm）强调把问题分解成子问题
面向对象范型（object-oriented paradigm）强调描述与问题相关的组件

3、过程单元：（procedure unit）
过程就是一系列指令，可以看成c语言中无返回值的函数，比较好理解.
简单来说，过程A中的过程B就是所谓的“过程单元”. A执行到B后暂停，执行B，执行完B后继续执行A
局部变量：在过程中定义的变量.
//形参是局部变量，因为它是在过程中定义.
形参、实参、按值传递、按引用传递：复习程设的时候一起复习了把

4、语法分析树，建议看fig5.18和fig5.19
//expression一定分解为term和expression，对term同样.
比如x+y分解成“x”，“+”和“y”，其中x是term，y是expression! 虽然它们看上去不太像。。
term x分解成x和“空”，x是factor，“空”是term
意思就是每分解一次，都会在最左端剥离出一个较小单元.

chap8
1、数组：
注意“数组多项式”，即把a[i][j]表示为a + n*i + j (有的时候数组从1开始计数）.
在数组中插入元素，需要将后面所有元素后移一个单位.
异构数组：可以理解为一个struct
//注意一个结构就是一个异构数组！当然前提是它的成员的数据类型不同.struct的存储方式，就是在内存里就是把它的成员一个一个排下去，但是各个成员占据各自的字节数，会各有不同。

2、表
什么时候链表为空？头指针指向NIL.

3、栈
书中栈指针指向栈顶元素，所以栈为空的标志是：栈指针指向栈底，即栈指针为首地址-1.
（但有时题目会给出，栈指针指向栈顶元素地址+1，根据题意处理）

4、队列
一个队列有头指针和尾指针，头指针指向首元素，尾指针指向末元素地址+1（见fig7.13)
添加元素：赋给尾指针指向的位置，尾指针+1；
提取元素：从头指针处读数据，删除源数据头指针+1；

5、二叉树：
注意了解一下不用指针存储二叉树的方法，逐层依次把根节点数据存在数组里.

chap11
1、图灵测试：
看一下定义吧，1950年由图灵提出.

2、8数码问题：
画宽度优先搜索树的时候，注意分配好空间.
建议做一下该章的习题，有一些问题变种.

chap12
1、图灵机：
按照表格一步一步来，很简单的啦，图灵机本来就是模仿人的计算.

（1）先根据当前的state和当前cell里的值，在表格里找到对应条目.
（2）根据表格里的要求，把当前cell的值修改了，接着左移或右移一格，之后改变state。这个state要记下来，下一次是要用的.

2、Barebones

十分简单，只有下面这几个语句，数据结构只有非负整数
clear（清零）
incr（自加）
decr（自减）（规定0自减还是0）
while x not 0 do（）end while
Barebones可以表示其他语句，包括if语句，乘法等，十分奇妙！详情可以看这一章的例子和习题.

extra：大事年表

最早的计算仪器：算盘
基于齿轮的设计的计算机器：Pascal，Leibniz，Babbage
巴贝奇的贡献：差分机，分析机（没钱完成）
Augusta Ada Byron 世界上第一个程序员（嫒~）

采用电子机械的计算机器：（被时代淘汰了）
George Stibitz，1940，Bell Laboratory
Mark I Harvard University
电子化的计算机器：（被沿用下来）
Atanasoff-Berry machine,1937-1941,John Atanasoff, Clifford Berry, Iowa State University
Colossus，建造于英国，用来在二战时破译德军密码

ENIAC（electronic numerical integral and calculator)
John Mauchly, J Presper Eckert, University of Pennsylvania
1946 十进制，程序和数据分离
EDVAC（elctronic discrete variable automatic computer)
1946 二进制，存储程序

Alan Turing 英国
1937年提出《论可计算数在判定问题中的应用》和图灵机
1950年提出图灵测试
1966年设立图灵奖

John Von Neumann 冯诺依曼 美籍匈牙利人 1903~1957 数学家，计算机科学家，物理学家，经济学家

Bool 布尔 1815~1864
1847年发表《An Investigation of the Laws of Thought》建立布尔代数