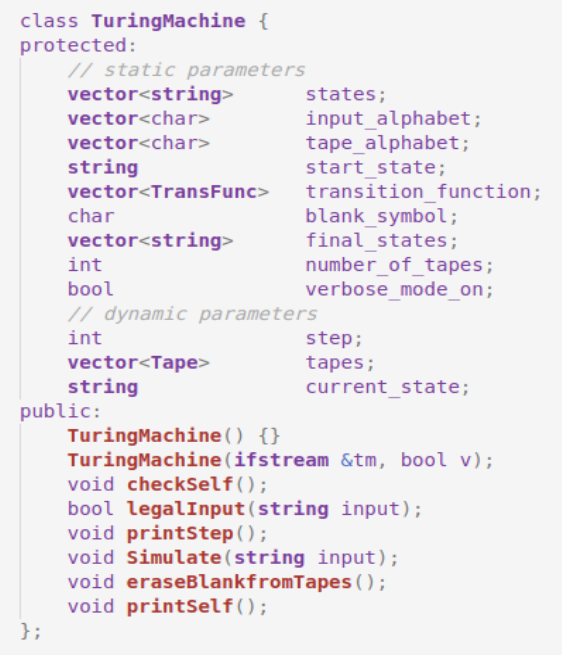
FLA (Fall 2021) – Project

Name: 张涵之 Dept: 计科

Grade: 2019 ID: 191220154

1. 分析与设计思路

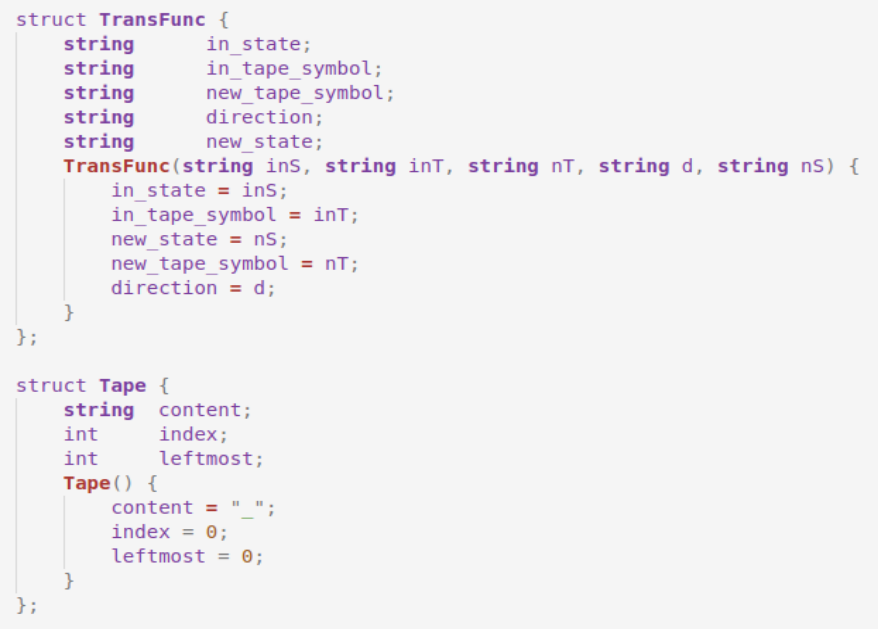
C++采用面向对象设计思想，因此首先设计图灵机的类。



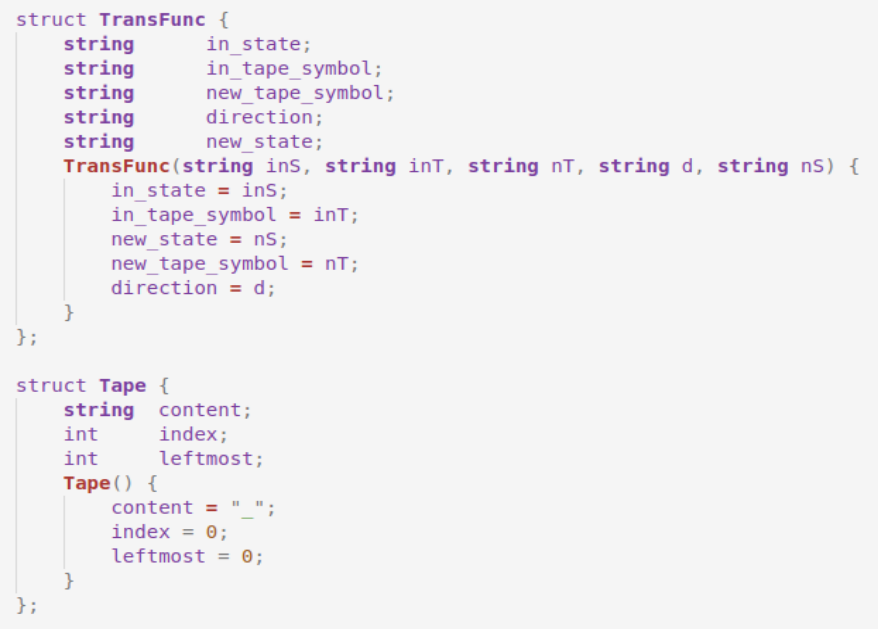
其中使用vector为容器表示集合，状态集由string表示，字符集用char，除图灵机程序语法中涉及的七部分内容，另有一bool变量表示图灵机的运行模式（是否开启verbose），以及运行过程中用到的step计数器、纸带集和当前所处状态标识。

该类的成员函数有从ifstream读文件初始化的构造函数（解析器），checkSelf用于初步构造完成后进一步检查语法，legalInput判断输入是否合法，printStep 在verbose模式下打印步骤，由Simulate（模拟器）调用。另有辅助函数eraseBlankfromTapes清除纸带两端多余的空白符，printSelf打印图灵机语法，主要在debug中使用。

设计转移函数，含新、旧状态，新、组符号组和方向，构造时从外部传入。



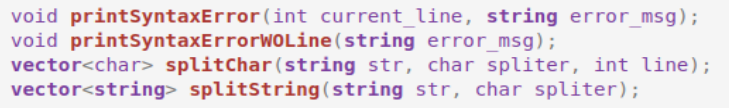
纸带类，含内容、当前index位置和最左端非空白符的位置，初始化为空。



几个辅助函数，其中printSyntaxError在图灵机类的构造函数中调用，一边读tm文件一边输出一些比较简单的语法错误（大多是类型错误，如赋值语句不完整，定义的状态名或符号使用了不合法的字符，缺少表示集合的花括号，输入的纸带数不是正整数等）。这类错误会结合在tm文件中出现的行号输出，方便定位。

printSyntaxErrorWOLine在图灵机类的checkSelf在调用，会检查定义的初始和终结状态、转移函数的新旧状态是否都在状态集中等“一致性”问题，并进行报错。

splitChar和splitString根据spliter（空格、逗号等）将整个字符串分割成单个字符或字符串集合，其中splitChar还会检查分割得到的是否全为单个字符。



接下来就是图灵机成员函数的具体实现了。

构造函数从文件的ifstream逐行读入，调用splitString和splitChar进行切分，确定每行是定义或注释，检查切分得到的字符串个数，是否有赋值符号等，若字符串数量多于预期，则判断多出的部分是否为注释，如下图片段展示：



checkSelf所做检查上面已经讨论过，这里不再赘述。

legalInput的逻辑也很简单，遍历字符串对其中每个字符在输入字符集中查找，若全都能找到则是合法输入，否则根据第一个不匹配字符的下标在verbose下输出。

printStep根据图灵机当前状态信息输出即可，只有一些细节，如为了显示美观而进行的空格对齐逐个让我比较头疼，在后面部分中会再详细谈到。

Simulate的逻辑是先做准备工作（设置当前步数为0，状态为初始状态，构造N条纸带并将输入复制到第一条上，此后采用while(true)无限循环，每轮先判断当前状态是否在终结状态集中，若是则跳出循环，否则开始查找和匹配转移函数（这里实现了通配符和一定程度上的优先级匹配，后面再详谈），匹配到之后开始更新纸带内容和移动index指针，设置新状态，调用eraseBlankfromTapes去掉纸带两端多余的空白符（以免造成打印时的赘余），更新完成，进入下一轮循环。

eraseBlankfromTapes和printSelf：比较白痴，不多说了，一些小bug后面再提。

接下来谈谈多带图灵机程序的编写，我采取了辗转相减法。使用三条纸带，初始状态下先遍历和清空第一条纸带，将两个整数分别拷贝到第二和第三条纸带上。

假设两个数分别为a和b，复制完成后，第二和第三条纸带的index都在这两个数的最右端。这时开始向前遍历，可见哪一条纸带先读到空白符，则它上面是更小的数字。不妨设更小的是第三条纸带上的b，则我们再从前往后遍历一次，一边删除纸带二a上的1，直到纸带三读到空白符，此时纸带二上剩余1的个数为a – b，然后让纸带二向前读到最右的1，此时纸带二和三的index又各自在a – b和b的最右端了。持续循环取“大数 = 大数 – 小数”直到某一次两条纸带自右向左同时读到

空白符，此时便得到最大公约数，再遍历一次将这个数复制回纸带一即可。

其实这个过程只用两条纸带也能做成，用三条主要是为了处理不合法的输入。根据这个图灵机的语法，合法输入的正则表达式为（11\*00\*11\*），即中间由一个或多个连续的零分割，两侧分别有且只有一个以上的1。对于任何不合法的输入，在“拷贝到纸带二和三”过程中，一经发现错误便停止复制，继续清空纸带一，并在上面填入错误信息“illegal”，填写完毕后进入终止状态halt\_reject，返回。





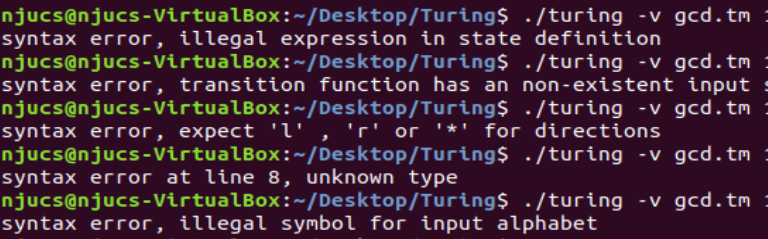
这里0为初始状态，cp1和cp2分别为拷贝两个数，sep为读中间的0，cmpr2l从右往左遍历比较两数，erase2和erase3通过逐个清除实现纸带二和三“相减”，getback实现相减后index回到所得差的最右端，illegal表示不合法输入，found表示已经找到gcd，illegal1-7输出“illegal”，halt\_accept和reject表示顺利完成或出错退出。

1. 实验完成度

完成了全部三个任务，其中解析器的错误处理对syntax error的具体类型进行了细化描述，额外输出行号和提示信息到stderr。模拟器支持普通和verbose模式，支持通配符\*，使用通配符情况下对多个转移函数匹配最精确的（\*最少的）。

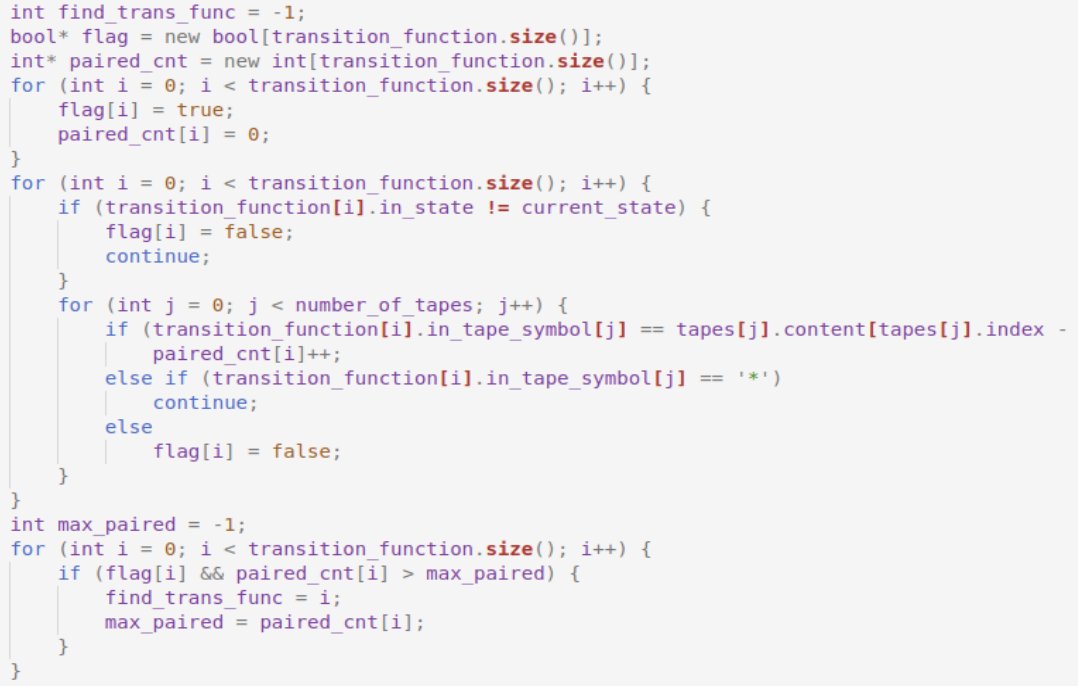
1. 实验中遇到的问题及解决方案

首先是各种异常情况（tm文件不存在，定义格式和逻辑错误，输入语法错误等）的定位、分类和报错，比较琐碎但难度不大，耐心即可。附上一些测试的输出：

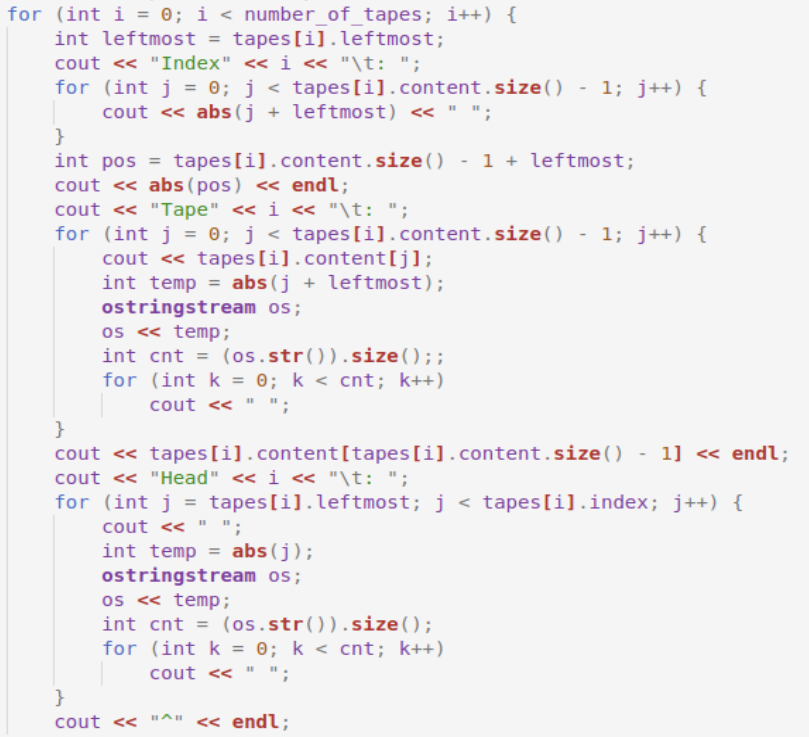


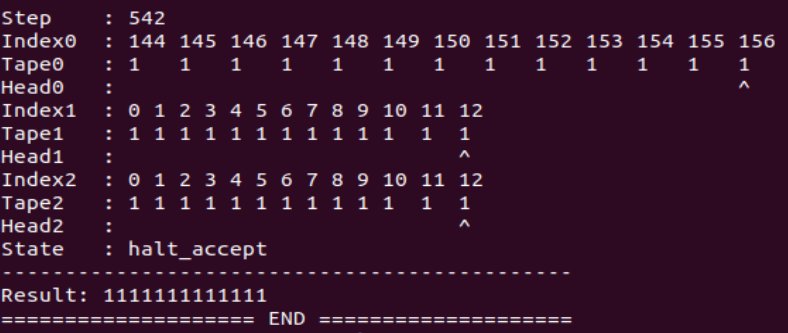
其次是纸带类的设计。我之前做的时候没有考虑到向0索引左边的纸带单元读写字符的情况，因此在调试过程中遇到一些麻烦。后来在index外添加了leftmost，对字符数组取下标时采用（index – leftmost）的方法，使纸带可以在左右两端“无限”扩容，比如左移超过当前leftmost范围时，对字符数组进行insert操作，leftmost递减即可。这样在删除两端多余空白符时，也可直接调用erase并使leftmost递增。这样实现了任何时刻纸带两端都没有多余的空白符，打印的时候也更加方便。

然后是通配符的实现。由于提供的示例程序含有通配符，且如果不实现精确匹配就会由于优先级不当出现死循环，因此我也实现了。大致逻辑是一个bool数组和一个int数组，分别标记每个转移函数是否能匹配，以及匹配的非通配符个数。遍历转移函数时更新这两个数组，完毕后选出能匹配且非通配符最多的函数即可。



最后是为了美观进行空格对齐……这里采用先取绝对值（省略负索引的符号）再将数字转成string，根据string的长度（即数字的位数）来决定输出多少个空格，实现不同位数的index，tape和head的对齐，很麻烦但是做完强迫症看着很爽。





这个是找了一个26和117的最大公约数是13的截图↑

这么看还挺慢的，辗转相除更快，但是感觉除法比减法难想，这次先算了~

1. 总结感想

还挺好玩的，就，手册说得很对，在纸上画得人头都大了，但是程序可以这么一步步地打印出来，看着就很清楚，写完感觉对图灵机的理解加深了o(\*￣▽￣\*)ブ

如果不是期末考试周到了还想自己多编几个测试样例玩一下的，不过时间有限就这样吧，也不知道这么点测试还有没有什么没发现的隐藏bug（s），嘤嘤嘤

1. 对课程和实验的意见与建议

实验非常好……没什么好建议了……但是课程的话这门课能不能多点课时，一周两次每次两节那种，一是下午连着上三节到后面真的又饿又困，尤其是冬天，那个空调一开，整个人都要冬眠过去了……二是上课能不能多讲几个（难一点的！）例子之类的，看课件经常觉得“嗯嗯，我已经懂了！”，等到看书或者写作业的时候，又发现“搞毛线啊，完全没懂……”说到这个，多出来的课时做习题课也蛮好的嘛，之前还没怎么上过这种全是作业课件英语，有时候题目都看不懂的课，有时候要到作业发下来了，才发现根本连题目意思都理解错了(╯‵□′)╯︵┻━┻