1. Try\_cath滤除
2. Tokenization
3. returnType MethodSignature MethodName [Parameters] MethodBody

注：关于returnType：

1.如果为构造函数，则为Constructor

2.识别不出来，则为varargs，或者known

3. 在方法签名前添加MethodSignature

//3.如果returnType经过Tokenization后，包含不止一个token，则将其合并成一个token.

如果在此合并，方法体中的相应部分也要进行合并。

1. 方法签名后，方法体前包含throw exception的情况，将其滤除
2. Comment中包含很多符合正则表达式[A-Za-z]+(\\.[A-Za-z]+)+形式的token，则在comment中出现的这些token，是看作单独的token处理的，而在代码中进行tokenization后，则分成了多个token，是不是将这些tokens合并成单个token更为合适，且能减少代码中vocabulary的词汇量 ？？？？
3. 单单经过第5步的处理不够，因为code中仍存在很多以java起始，符合[A-Za-z]+([\\.[A-Za-z](\\\\.[A-Za-z)]+)+形式的tokens，所以进行了进一步的处理，将java代码中，符合此约束的tokens合并成单个的token，从而可以一定程度上降低输入的长度，以及词汇表的大小。
4. comment中还有类似 #MethodSignature 形式的内容，应该怎么处理？？？？？
5. 如何对方法名进行处理？？？？？？
6. 如何对对应的comment进行处理？？？？？？？？？？？？？？
7. 代码中会含有类似于thisonFinished...的片段，预处理的时候需要注意
8. 将代码中的for 和 if 结构进行了重构
9. 代码的参数中包含很多类似于@NotNull之类的信息，所以将其滤除是必要的
10. 代码地方法体中包含一些类似于@SuppressWarnings()之类的信息，将其滤除也是有必要的
11. EnhancedFor 和 For不同。
12. Try()语句和try-catch的不同
13. 原来真的会出现if 超过一百次的情况