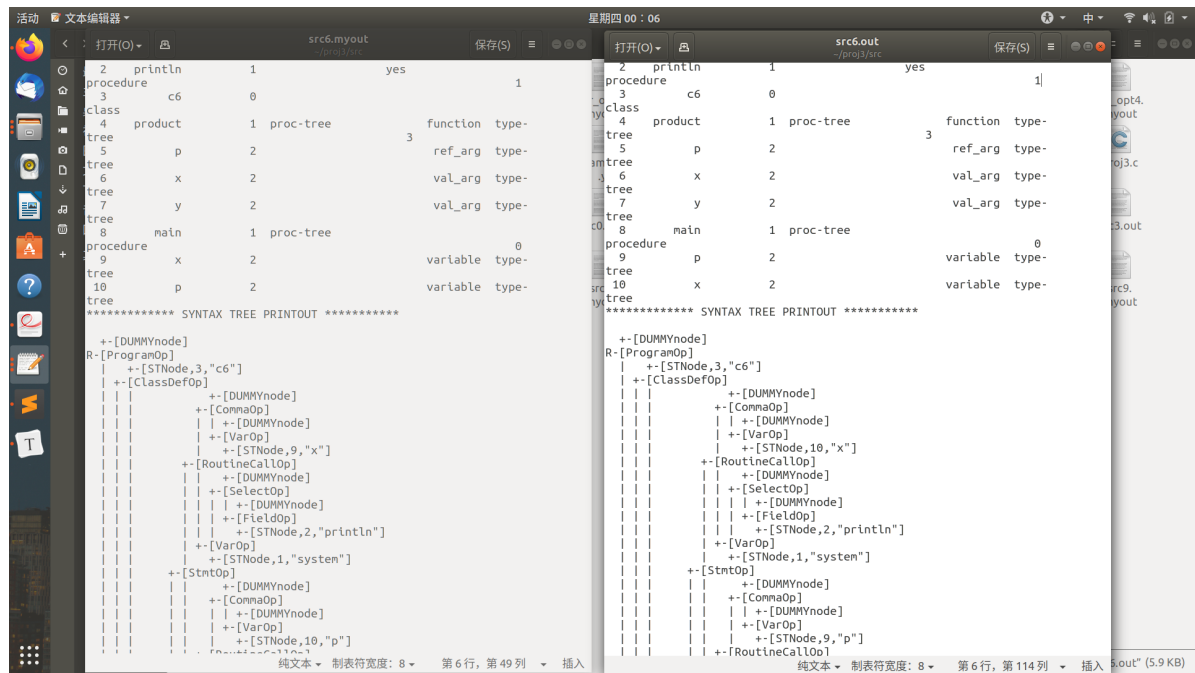


可以看到在截图中间位置（在 [NumNode,7] 往上两行）我的多出了两行 [DUMMYNode]、[BodyOp]。

3.有一些ID在符号表中的顺序不一样，比如：



```
src6.myout
2  println  1  yes  1
procedure
3  c6  0
class
4  product  1  proc-tree  function type-
tree
5  p  2  ref_arg type-
tree
6  x  2  val_arg type-
tree
7  y  2  val_arg type-
tree
8  main  1  proc-tree
procedure
9  x  2  variable type-
tree
10 p  2  variable type-
tree
***** SYNTAX TREE PRINTOUT *****
+-[DUMMYNode]
R-[ProgramOp]
| +-[STNode,3,"c6"]
| +-[ClassDefOp]
| | +-[DUMMYNode]
| | +-[CommaOp]
| | | +-[DUMMYNode]
| | | +-[VarOp]
| | | +-[STNode,9,"x"]
| | +-[RoutineCallOp]
| | +-[DUMMYNode]
| | +-[SelectOp]
| | | +-[DUMMYNode]
| | | +-[FieldOp]
| | | +-[STNode,2,"println"]
| | +-[VarOp]
| | +-[STNode,1,"system"]
| +-[StntOp]
| | +-[DUMMYNode]
| | +-[CommaOp]
| | | +-[DUMMYNode]
| | | +-[VarOp]
| | | +-[STNode,10,"p"]
| | +-[RoutineCallOp]

src6.out
2  println  1  yes  1
procedure
3  c6  0
class
4  product  1  proc-tree  function type-
tree
5  p  2  ref_arg type-
tree
6  x  2  val_arg type-
tree
7  y  2  val_arg type-
tree
8  main  1  proc-tree
procedure
9  x  2  variable type-
tree
10 p  2  variable type-
tree
***** SYNTAX TREE PRINTOUT *****
+-[DUMMYNode]
R-[ProgramOp]
| +-[STNode,3,"c6"]
| +-[ClassDefOp]
| | +-[DUMMYNode]
| | +-[CommaOp]
| | | +-[DUMMYNode]
| | | +-[VarOp]
| | | +-[STNode,10,"x"]
| | +-[RoutineCallOp]
| | +-[DUMMYNode]
| | +-[SelectOp]
| | | +-[DUMMYNode]
| | | +-[FieldOp]
| | | +-[STNode,2,"println"]
| | +-[VarOp]
| | +-[STNode,1,"system"]
| +-[StntOp]
| | +-[DUMMYNode]
| | +-[CommaOp]
| | | +-[DUMMYNode]
| | | +-[VarOp]
| | | +-[STNode,9,"p"]
| | +-[RoutineCallOp]
```

注意到最后两个符号x、p顺序相反了，因此在语法树上 STNode 节点对应的索引值也会反过来，但标准答案与我的输出都会保证语法树上的索引始终与符号表的索引保持一致。因此这并不算是错误。