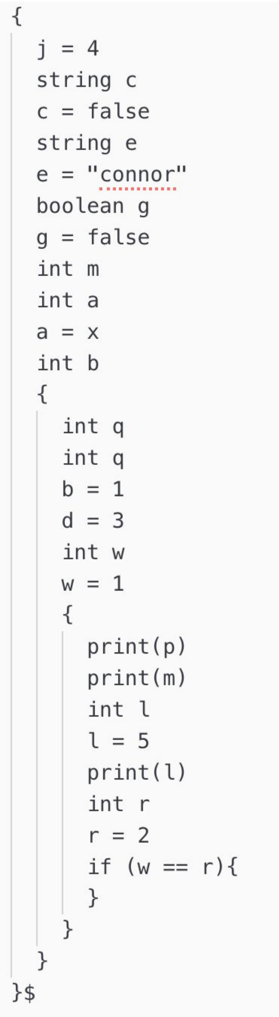
Pure Python LL1 Complier. Run python3 custom.txt <code.txt>

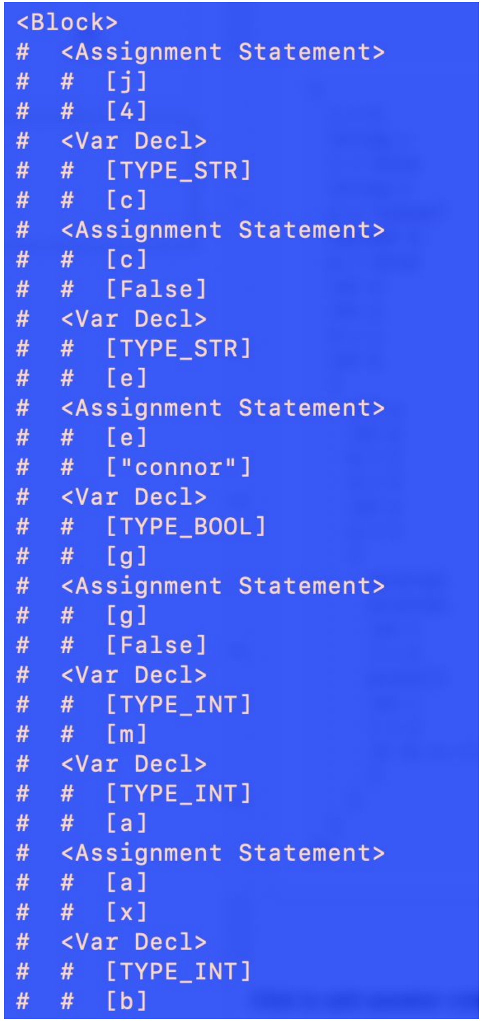
P3

Connor Gillis

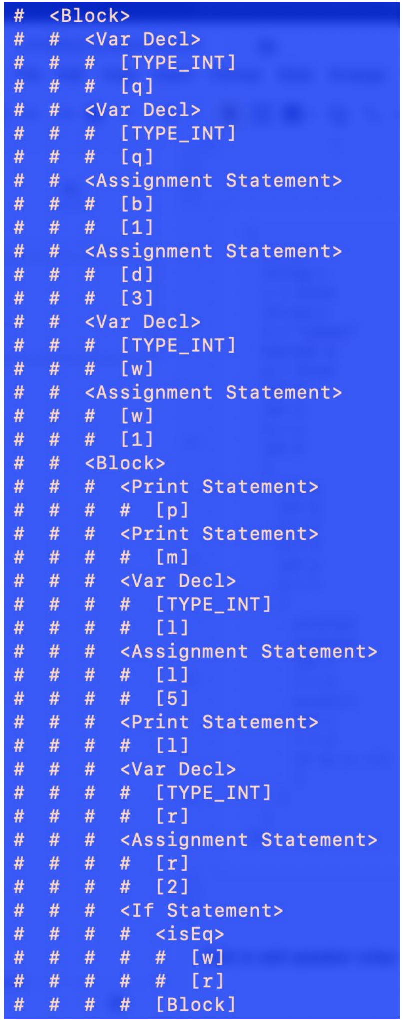


```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

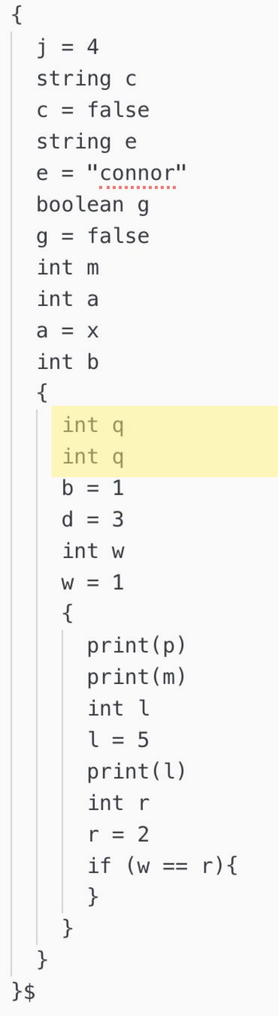
SC => AST



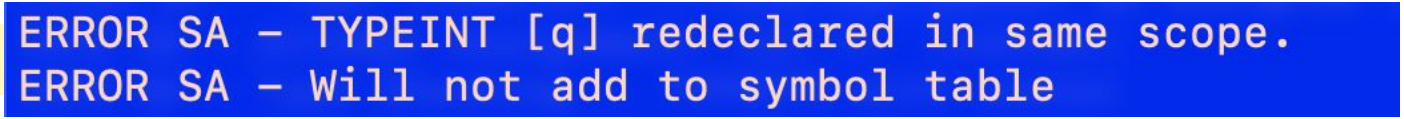
```
<Block>
#  <Assignment Statement>
#  #  [j]
#  #  [4]
#  <Var Decl>
#  #  [TYPE_STR]
#  #  [c]
#  <Assignment Statement>
#  #  [c]
#  #  [False]
#  <Var Decl>
#  #  [TYPE_STR]
#  #  [e]
#  <Assignment Statement>
#  #  [e]
#  #  ["connor"]
#  <Var Decl>
#  #  [TYPE_BOOL]
#  #  [g]
#  <Assignment Statement>
#  #  [g]
#  #  [False]
#  <Var Decl>
#  #  [TYPE_INT]
#  #  [m]
#  <Var Decl>
#  #  [TYPE_INT]
#  #  [a]
#  <Assignment Statement>
#  #  [a]
#  #  [x]
#  <Var Decl>
#  #  [TYPE_INT]
#  #  [b]
```



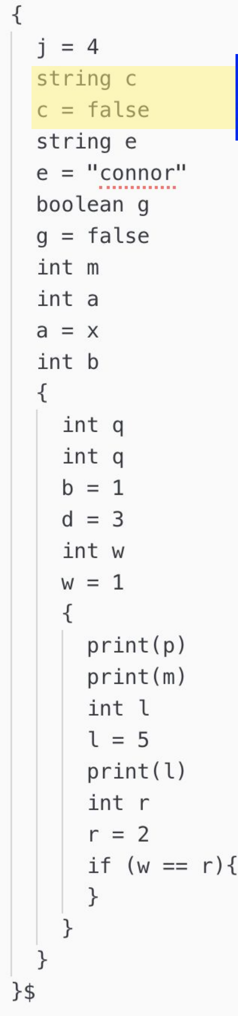
```
#  <Block>
#  #  <Var Decl>
#  #  #  [TYPE_INT]
#  #  #  [q]
#  #  <Var Decl>
#  #  #  [TYPE_INT]
#  #  #  [q]
#  #  <Assignment Statement>
#  #  #  [b]
#  #  #  [1]
#  #  <Assignment Statement>
#  #  #  [d]
#  #  #  [3]
#  #  <Var Decl>
#  #  #  [TYPE_INT]
#  #  #  [w]
#  #  <Assignment Statement>
#  #  #  [w]
#  #  #  [1]
#  #  <Block>
#  #  #  <Print Statement>
#  #  #  #  [p]
#  #  #  <Print Statement>
#  #  #  #  [m]
#  #  #  <Var Decl>
#  #  #  #  [TYPE_INT]
#  #  #  #  [l]
#  #  #  <Assignment Statement>
#  #  #  #  [l]
#  #  #  #  [5]
#  #  #  <Print Statement>
#  #  #  #  [l]
#  #  #  <Var Decl>
#  #  #  #  [TYPE_INT]
#  #  #  #  [r]
#  #  #  <Assignment Statement>
#  #  #  #  [r]
#  #  #  #  [2]
#  #  #  <If Statement>
#  #  #  #  <isEq>
#  #  #  #  #  [w]
#  #  #  #  #  [r]
#  #  #  #  [Block]
```



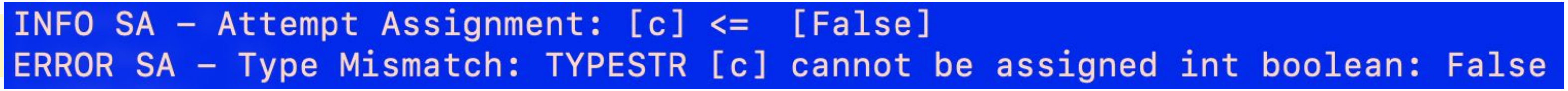
```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```



```
ERROR SA – TYPEINT [q] redeclared in same scope.
ERROR SA – Will not add to symbol table
```

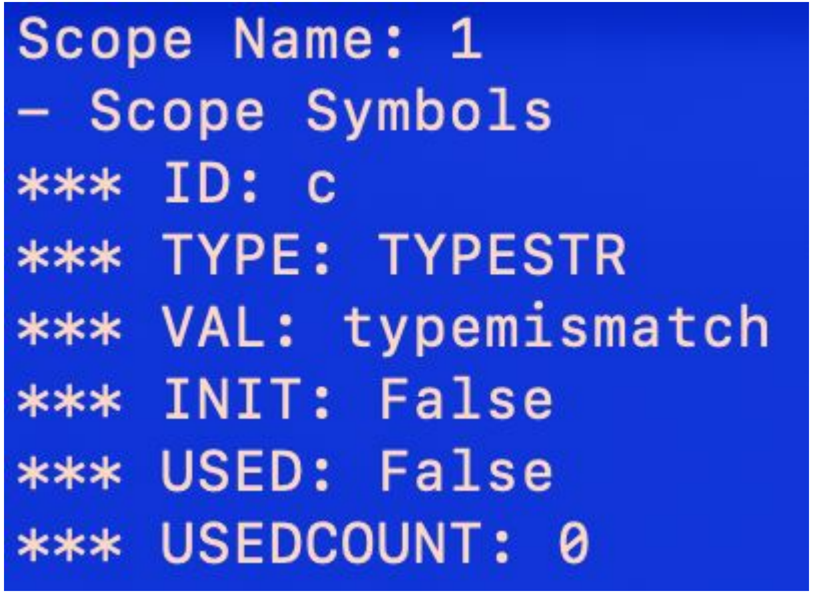


```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

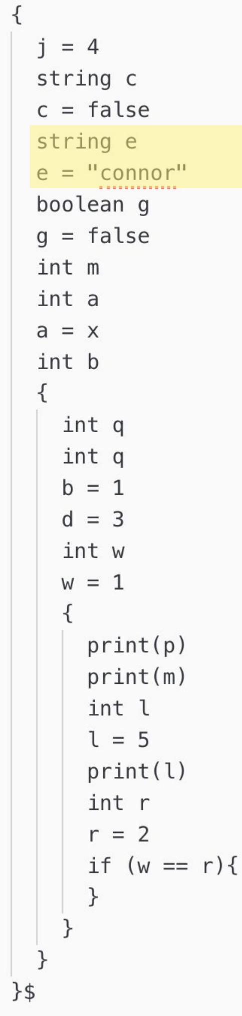


```
INFO SA - Attempt Assignment: [c] <=  [False]
ERROR SA - Type Mismatch: TYPESTR [c] cannot be assigned int boolean: False
```

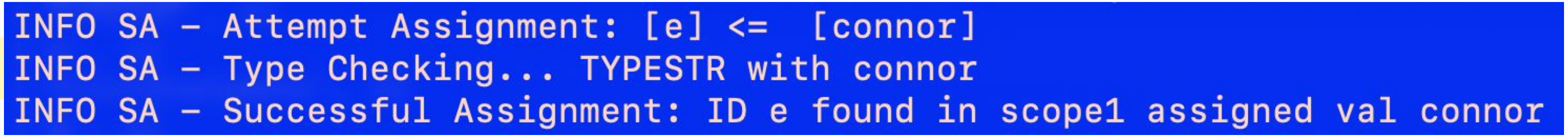
Type mismatch on assign, don’t set init to true



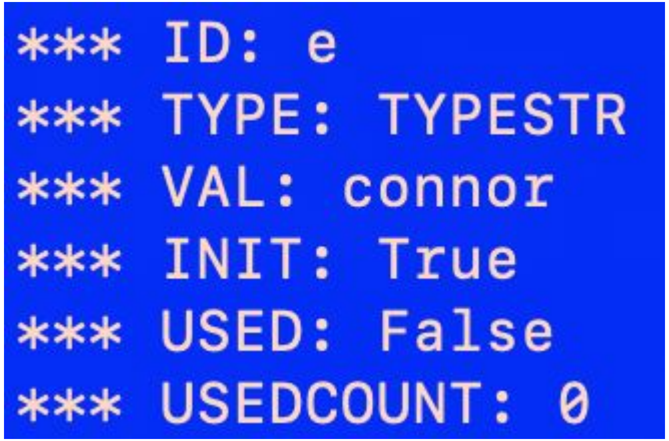
```
Scope Name: 1
- Scope Symbols
*** ID: c
*** TYPE: TYPESTR
*** VAL: typemismatch
*** INIT: False
*** USED: False
*** USEDCOUNT: 0
```



```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

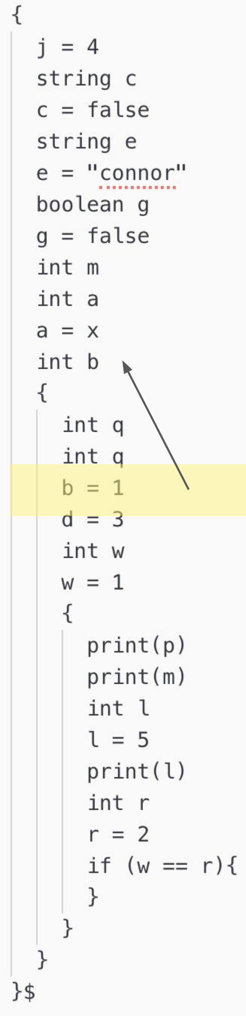


```
INFO SA - Attempt Assignment: [e] <=  [connor]
INFO SA - Type Checking... TYPESTR with connor
INFO SA - Successful Assignment: ID e found in scope1 assigned val connor
```

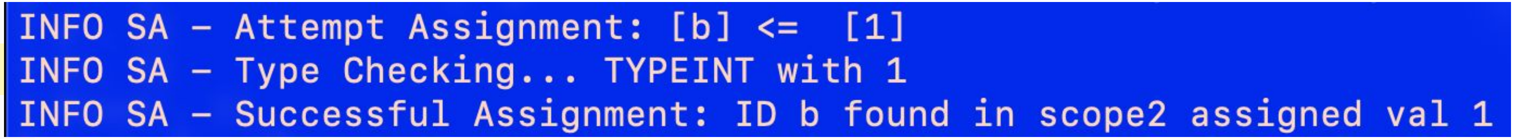


```
*** ID: e
*** TYPE: TYPESTR
*** VAL: connor
*** INIT: True
*** USED: False
*** USEDCOUNT: 0
```

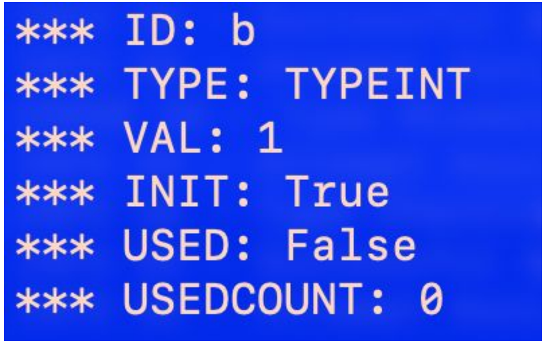
No type mismatch on assign, set init to true



```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

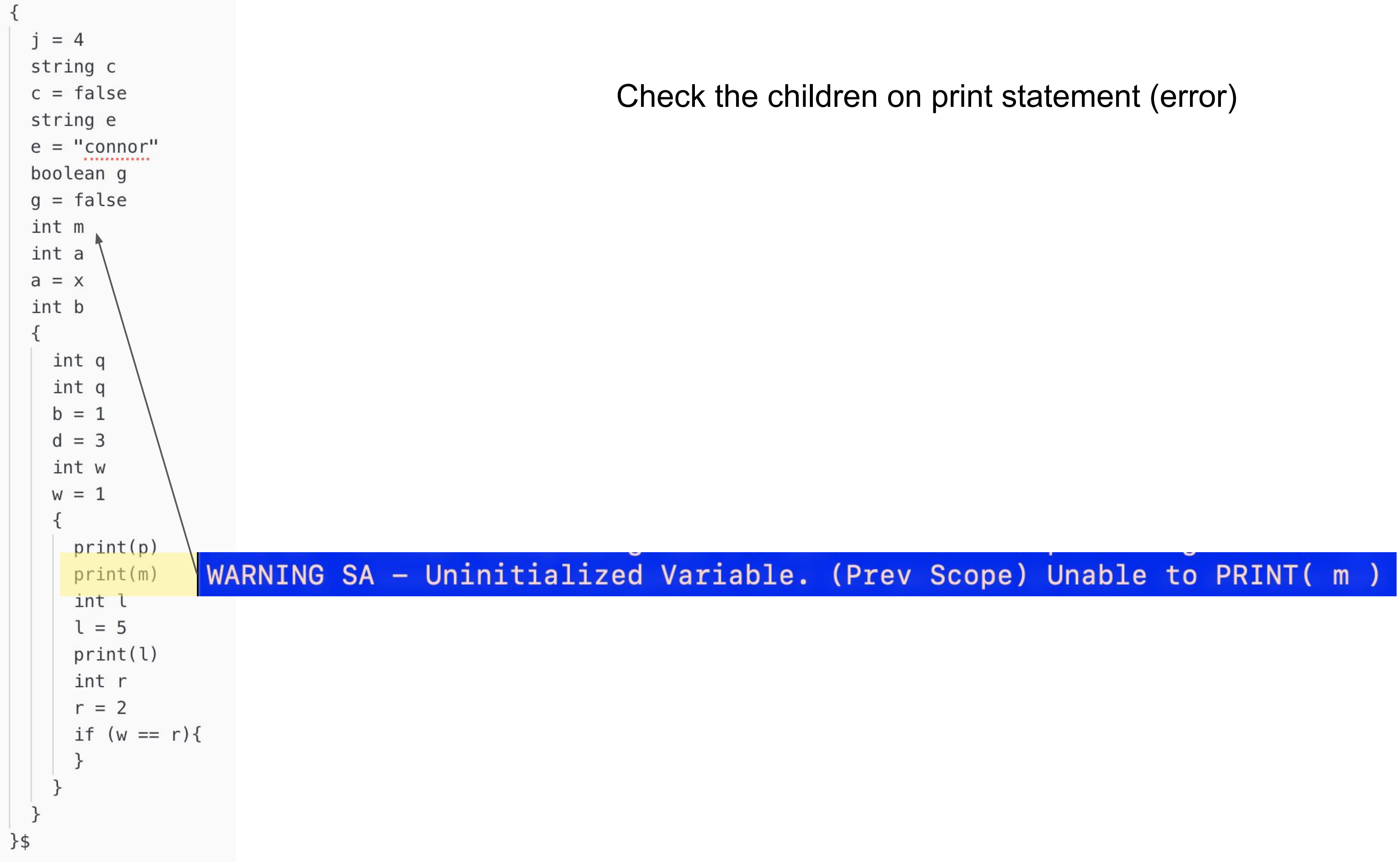


```
INFO SA - Attempt Assignment: [b] <=  [1]
INFO SA - Type Checking... TYPEINT with 1
INFO SA - Successful Assignment: ID b found in scope2 assigned val 1
```



```
*** ID: b
*** TYPE: TYPEINT
*** VAL: 1
*** INIT: True
*** USED: False
*** USEDCOUNT: 0
```

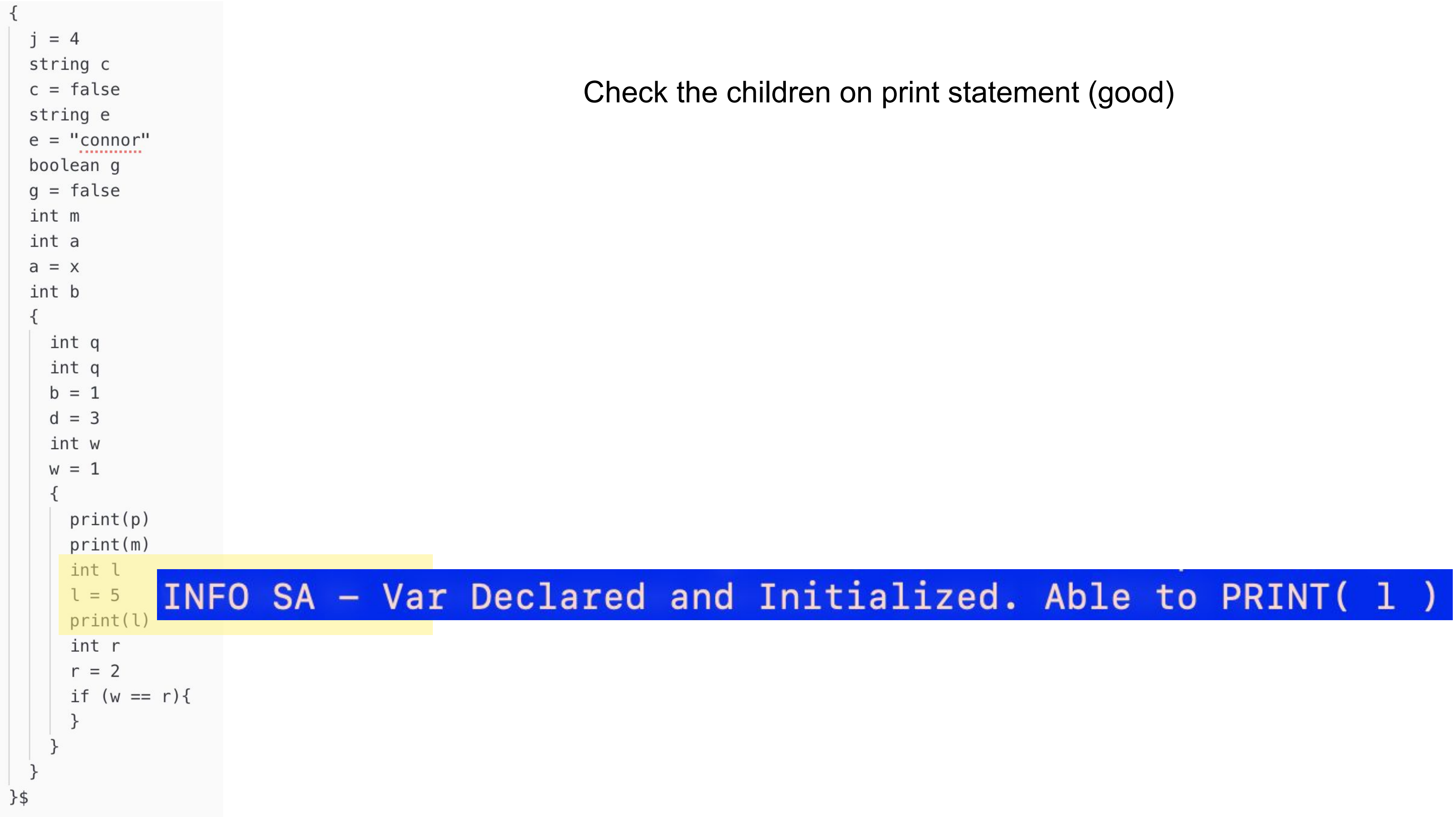
Check the children on print statement (error)



```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

```
WARNING SA - Uninitialized Variable. (Prev Scope) Unable to PRINT( m )
```

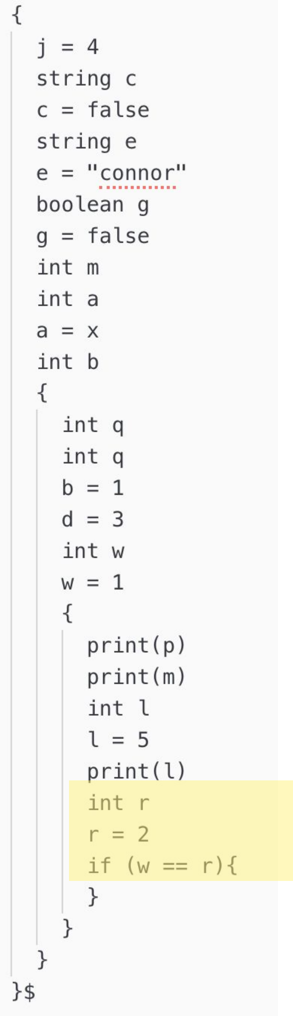
Check the children on print statement (good)



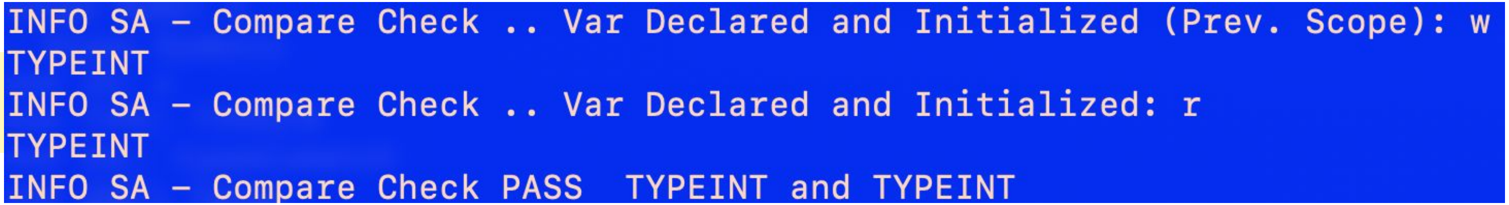
```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```

```
INFO SA – Var Declared and Initialized. Able to PRINT( l )
```

Check type compatibility for isEq and isNotEq children



```
{
  j = 4
  string c
  c = false
  string e
  e = "connor"
  boolean g
  g = false
  int m
  int a
  a = x
  int b
  {
    int q
    int q
    b = 1
    d = 3
    int w
    w = 1
    {
      print(p)
      print(m)
      int l
      l = 5
      print(l)
      int r
      r = 2
      if (w == r){
      }
    }
  }
}$
```



```
INFO SA – Compare Check .. Var Declared and Initialized (Prev. Scope): w
TYPEINT
INFO SA – Compare Check .. Var Declared and Initialized: r
TYPEINT
INFO SA – Compare Check PASS  TYPEINT and TYPEINT
```