import unittest

from TestUtils import TestParser

class ParserSuite(unittest.TestCase):

    def test\_1(self):

        input = """ fact: function integer (n: integer) {

            for (i=1, i<10,i+1) {

                a= {1,2};

            {

               a: integer;

            }

            }

        } """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 201))

    def test\_2(self):

        input = """ fact: function integer (n: integer) {

            for (i=1, i<10,i+1) {

                a= 2==3;

            {

               return;

            }

            }

        } """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 202))

    def test\_3(self):

        input = """main: function void () {}"""

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 203))

    def test\_4(self):

        input = """x: integer = 65;

        fact: function integer (n: integer) {

            if (n == 0) return 1;

            else return n \* fact(n - 1);

        }

        inc: function void(out n: integer, delta: integer) {

            n = n + delta;

        }

        main: function void() {

            delta: integer = fact(3);

            inc(x, delta);

            printInteger(x);

        }"""

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 204))

    def test\_5(self):

        input = """a, b, c, d: integer = 3, 4, 6;"""

        expect = "Error on line 1 col 29: ;"

        self.assertTrue(TestParser.test(input, expect, 205))

    def test\_6(self):

        input = """a, b, c: integer = 3, 4, 6;"""

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 206))

    def test\_7(self):

        input = """if (n == 0) return 1;"""

        expect = "Error on line 1 col 0: if"

        self.assertTrue(TestParser.test(input, expect, 207))

    def test\_8(self):

        input = """a : array [2,3] of integer = {1,2,3};"""

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 208))

    def test\_9(self):

        input = """a : integer = {1,2,3};"""

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 209))

    def test\_10(self):

        input = input = """x: integer = 65;

        fact: function integer (n: integer) {

            if (n == 0) return 1;

            else return n \* fact(n - 1);

            {

            }

        }

        inc: function void(out n: integer, delta: integer) {

            n = n + delta;

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 210))

    def test\_11(self):

        input = input = """x: integer = 65;

        fact: function integer (n: integer) {

            if (n == 0) return 1;

            else return n \* fact(n - 1);

            {

                ;

            }

        }

        inc: function void(out n: integer, delta: integer) {

            n = n + delta;

        }

        """

        expect = "Error on line 7 col 16: ;"

        self.assertTrue(TestParser.test(input, expect, 211))

    def test\_12(self):

        input = input = """x: integer = 65;

        fact: function integer (n: integer) {

            if (n == 0) return 1;

            else return n \* fact(n - 1);

            {

            }

        }

        inc: function void(out n: integer, delta: integer) inherit abc {

            n = n + delta;

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 212))

    def test\_13(self):

        input = input = """x: integer = 65;

        fact: function integer (n: integer) {

            if (n == 0) return 1;

            else return n \* fact(n - 1);

            {

            }

        }

        inc: function void(out n: integer, delta: integer) inherit abc {

            n = abc(2);

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 213))

    def test\_14(self):

        input = """

        inc: function void(out n: integer, delta: integer) inherit abc {

            n = (2+3);

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 214))

    def test\_15(self):

        input = """

        inc: function void(out n: integer, delta: integer) inherit abc {

            n = a[2==3,a>3];

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 215))

    def test\_16(self):

        input = """

        inc: function void(out n: integer, delta: integer) inherit abc {

            n = readBoolean() ;

        }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 216))

    def test\_17(self):

        input = """

        a:integer;

          fact: function integer (){

              do {x =2 ;}

              while(true);

              a=2;

              a: integer;

              return "123";

              a=2;

              return "123";

          }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 217))

    def test\_18(self):

        input = """

        a:integer;

          fact: function integer (){

              do {x = {1,2,3} ;}

              while (true);

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 218))

    def test\_19(self):

        input = """

        a:integer;

          fact: function integer (){

              do {x = {1,2,3,.123e+10} ;}

              while (true);

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 219))

    def test\_20(self):

        input = """

        a:integer;

          fact: function integer (){

              do {x = {1,2,3,01} ;}

              while (true);

            }

        """

        expect = "Error on line 4 col 30: 1"

        self.assertTrue(TestParser.test(input, expect, 220))

    def test\_21(self):

        input = """

        a:integer=1;

          fact: function integer (){

              do {x = {1,2,3,01} ;}

              while (true);

            }

        """

        expect = "Error on line 4 col 30: 1"

        self.assertTrue(TestParser.test(input, expect, 221))

    def test\_22(self):

        input = """

        a:integer;

          fact: function integer (){

              a = 1;

              do {x = {1,2,3,1} ;}

              while (true);

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 222))

    def test\_23(self):

        input = """

        a:integer = a[1];

          fact: function integer (){

              a = 1;

              do {x = {1,2,3,1} ;}

              while (true);

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 223))

    def test\_24(self):

        input = """

        a:integer = a[1];

          fact: function integer (){

              {}

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 224))

    def test\_25(self):

        input = """

        a:integer = a[1];

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 225))

    def test\_26(self):

        input = """

        a:integer = a[1];

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 226))

    def test\_27(self):

        input = """

        a:integer = a[1];

          fact: function integer (){

              {

                  {

                  return {12\_3.e+5,123,"he ask me \\"",true, a[1,2],foo()};

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 227))

    def test\_28(self):

        input = """

        a:integer = "he aske me\n";

          fact: function integer (){

              {

                  {

                  return {12\_3.e+5,123,"he ask me \\"",true, a[1,2],foo()};

                  }

              }

            }

        """

        expect = "he aske me"

        self.assertTrue(TestParser.test(input, expect, 228))

    def test\_29(self):

        input = """

        a:integer = truee;

          fact: function integer (){

              {

                  {

                  return {12\_3.e+5,123,"he ask me \\"",true, a[1,2],foo()};

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 229))

    def test\_30(self):

        input = """

        a:integer = truee;

          fact: function integer (){

              {

                  {

                  return ;

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 230))

    def test\_31(self):

        input = """

        a:integer = truee;

          fact: function integer (){

              {

                  {

                  if(x==2);

                  else ;

                  }

              }

            }

        """

        expect = "Error on line 6 col 26: ;"

        self.assertTrue(TestParser.test(input, expect, 231))

    def test\_32(self):

        input = """

        a:integer = truee;

          fact: function integer (){

              {

                  {

                  if(x==2);

                  else ;

                  }

              }

            }

        """

        expect = "Error on line 6 col 26: ;"

        self.assertTrue(TestParser.test(input, expect, 232))

    def test\_33(self):

        input = """

        a:integer = 1 ;

          fact: function integer (){

              {

                  {

                  if(x==2) a=2;

                  else a=2;

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 233))

    def test\_34(self):

        input = """

        a,b,c:integer =1 ;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "Error on line 2 col 25: ;"

        self.assertTrue(TestParser.test(input, expect, 234))

    def test\_35(self):

        input = """

        a,b,c:integer =1,a[2[3]],3;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "Error on line 2 col 28: ["

        self.assertTrue(TestParser.test(input, expect, 235))

    def test\_36(self):

        input = """

        a,b,c:integer =1,a[a[3]],3;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 236))

    def test\_37(self):

        input = """

        a,b,c:integer =1,a[a[3]],3;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 237))

    def test\_38(self):

        input = """

        a,b,c:array [2\_1\_2,3\_1] of string;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 238))

    def test\_39(self):

        input = """

        a,b,c:array [2\_1\_2,3\_] of string;

          fact: function integer (){

              {

                  {

                  }

              }

            }

        """

        expect = "Error on line 2 col 28: \_"

        self.assertTrue(TestParser.test(input, expect, 239))

    def test\_40(self):

        input = """

        a,b,c:array [2\_1\_2,3] of string;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 240))

    def test\_41(self):

        input = """

        a,b,c: auto;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 241))

    def test\_42(self):

        input = """

        a,b,c: void;

        """

        expect = "Error on line 2 col 15: void"

        self.assertTrue(TestParser.test(input, expect, 242))

    def test\_43(self):

        input = """

        \_a\_,b,c: string ;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 243))

    def test\_44(self):

        input = """

        \_a\_,0b,c: string ;

        """

        expect = "Error on line 2 col 12: 0"

        self.assertTrue(TestParser.test(input, expect, 244))

    def test\_45(self):

        input = """

        \_a\_,\_0b,c: boolean ;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 245))

    def test\_46(self):

        input = """

        \_a\_,\_0b,c: float;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 246))

    def test\_47(self):

        input = """

        \_a\_,\_0b,c: float;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 247))

    def test\_48(self):

        input = """

        /\* 0a:integer;

        \*/

        a: integer;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 248))

    def test\_49(self):

        input = """

        // 0a:integer;

        a: integer;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 249))

    def test\_50(self):

        input = """

        // 0a:integer;

        a: integer;

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 250))

    def test\_51(self):

        input = """

        // 0a:integer;

        {}

        """

        expect = "Error on line 4 col 8: {"

        self.assertTrue(TestParser.test(input, expect, 251))

    def test\_52(self):

        input = """

        a:integer = a[1];

          fact: function integer (inherit out x:integer){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 252))

    def test\_53(self):

        input = """

        a:integer = a[1];

          fact: function integer (inherit x:integer){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 253))

    def test\_54(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:integer){

              {

                  {

                  }

              }

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 254))

    def test\_55(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:void){

              {

                  {

                  }

              }

            }

        """

        expect = "Error on line 3 col 40: void"

        self.assertTrue(TestParser.test(input, expect, 255))

    def test\_55(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:void){

              {

                  {

                  }

              }

            }

        """

        expect = "Error on line 3 col 40: void"

        self.assertTrue(TestParser.test(input, expect, 255))

    def test\_56(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              if(x==2) a=b[2,2==3];

              if(x>3) a=2;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 256))

    def test\_57(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              if(x==2) a=b[2,foo(2)];

              if(x>3) a=2;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 257))

    def test\_58(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              if(x==2) a=b[2,foo(x,2),c[1]];

              if(x>3) a=2;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 258))

    def test\_59(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              if(x==2) a=b[2,foo(x,2),c[1]];

              if(x>3) a=2;

              a= 2==3;

              a= (2+3);

              for(arr[1]=0, arr[1]<10,arr[1]+1);

              while(x<2)x=x+1;

            }

        """

        expect = "Error on line 8 col 47: ;"

        self.assertTrue(TestParser.test(input, expect, 259))

    def test\_60(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              for (i = 1, i < 10, i + 1) {

                writeInt(i);

                {

                    ;

                }

            }

            }

        """

        expect = "Error on line 7 col 20: ;"

        self.assertTrue(TestParser.test(input, expect, 260))

    def test\_61(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              for (i = 1, i < 10, i + 1) {

                writeInt(i);

                {

                    ;

                }

            }

            }

        """

        expect = "Error on line 7 col 20: ;"

        self.assertTrue(TestParser.test(input, expect, 261))

    def test\_62(self):

        input = """

        a:integer = a[1];

        return ;

          fact: function integer (out x:string){

              for (i = 1, i < 10, i + 1) {

                writeInt(i);

                {

                    ;

                }

            }

            }

        """

        expect = "Error on line 3 col 8: return"

        self.assertTrue(TestParser.test(input, expect, 262))

    def test\_63(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              {

            }

        """

        expect = "Error on line 8 col 8: <EOF>"

        self.assertTrue(TestParser.test(input, expect, 263))

    def test\_64(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              while(x==2) a=3;}

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 264))

    def test\_65(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              do ;

              while (x>=2);

            }

        """

        expect = "Error on line 4 col 17: ;"

        self.assertTrue(TestParser.test(input, expect, 265))

    def test\_66(self):

        input = """

        a:integer = a[1];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 266))

    def test\_67(self):

        input = """

        a:integer = a[1,{1,2,3}];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 267))

    def test\_68(self):

        input = """

        a:integer = a[1,/\*111\_\*/2];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 268))

    def test\_69(self):

        input = """

        a:integer = a[1,/\*111\_\*/2\_];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 33: \_"

        self.assertTrue(TestParser.test(input, expect, 269))

    def test\_70(self):

        input = """

        a:integer = a[1,2/\*111\_\*///aa];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 3 col 10: fact"

        self.assertTrue(TestParser.test(input, expect, 270))

    def test\_71(self):

        input = """

        a:integer = a[1,foo(1,{1,2,3}),boolean];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 39: boolean"

        self.assertTrue(TestParser.test(input, expect, 271))

    def test\_72(self):

        input = """

        a:integer = a[1,foo(1,{1,2,3}),integer];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 39: integer"

        self.assertTrue(TestParser.test(input, expect, 272))

    def test\_73(self):

        input = """

        a:integer = a[1,foo(1,{1,2,3}),float];

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 39: float"

        self.assertTrue(TestParser.test(input, expect, 273))

    def test\_74(self):

        input = """

        a:integer = aa :: bb;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 274))

    def test\_75(self):

        input = """

        a:integer = aa :: bb :: cc;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 29: ::"

        self.assertTrue(TestParser.test(input, expect, 275))

    def test\_76(self):

        input = """

        a:integer = (aa :: bb) :: cc;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 276))

    def test\_77(self):

        input = """

        a:integer = (aa == bb) >= cc;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 277))

    def test\_78(self):

        input = """

        a:integer = aa == bb >= cc;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "Error on line 2 col 29: >="

        self.assertTrue(TestParser.test(input, expect, 278))

    def test\_79(self):

        input = """

        a:integer = (aa && bb)&&cc;

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 279))

    def test\_80(self):

        input = """

        a:integer = 1 && (bb&&cc);

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 280))

    def test\_81(self):

        input = """

        a:integer = {foo(1,2),2};

          fact: function integer (out x:string){

              a=a[2,3];

              b= {1,2,3};

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 281))

    def test\_82(self):

        input = """

        a : boolean = true;

                add: function void(i: float) {

                    return i;

                }

                inc: function void(out n: integer, inherit delta: integer, inherit out hallo: auto) inherit add{

                    n = n + delta + hallo;

                    n = n - delta - hallo;

                    n = n \* delta \* hallo;

                    /\* multiple

                        line

                        comment

                    \*/

                    n = a[1, a[1, a[1,2]]]; // n[1,2] = 2 => n = a[1,2];

                    return n;

        """

        expect = "Error on line 16 col 8: <EOF>"

        self.assertTrue(TestParser.test(input, expect, 282))

    def test\_83(self):

        input = """

        a : boolean = true;

                add: function void(i: float) {

                    return i;

                }

                inc: function void(out n: integer, inherit delta: integer, inherit out hallo: auto) inherit add{

                    n = n + delta + hallo;

                    n = n - delta - hallo;

                    n = n \* delta \* hallo;

                    /\* multiple

                        line

                        comment

                    \*/

                    n = a[1, a[1, a[1,2]]]; // n[1,2] = 2 => n = a[1,2];

                    return n;

                    }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 283))

    def test\_84(self):

        input = """

        a : boolean = true;

                add: function void(i: float) {

                    return i;

                }

                inc: function void(out n: integer, inherit delta: integer, inherit out hallo: auto) inherit add{

                    n = n + delta + hallo;

                    return ;

                    }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 284))

    def test\_85(self):

        input = """

        a : boolean = true;

                add: function void(i: float) {

                    return i;

                }

                inc: function void(out n: integer, inherit delta: integer, inherit out hallo: auto) inherit add{

                    n = n + delta + hallo;

                    return ;

                    ;

                    }

        """

        expect = "Error on line 9 col 20: ;"

        self.assertTrue(TestParser.test(input, expect, 285))

    def test\_86(self):

        input = """

        learningtofly: function integer() {

                for (i = -foo(), i < 10, i :: -2 :: 3E-10) {

                    a = ---a;

                    }

        }

        """

        expect = "Error on line 3 col 49: ::"

        self.assertTrue(TestParser.test(input, expect, 286))

    def test\_87(self):

        input = """

        main: function void () {

           for (i = 1, i < 10, i + 1) {

                writeInt(i);

                }

                return "aa"::"bb";

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 287))

    def test\_88(self):

        input = """

            main:function void()

            {

                a:string="aaa"::"bbb"=="ccc";

                return;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 288))

    def test\_89(self):

        input = """

            main:function void()

            {

                a:string="aaa"::"bbb"::"ccc";

                return;

            }

        """

        expect = "Error on line 4 col 37: ::"

        self.assertTrue(TestParser.test(input, expect, 289))

    def test\_90(self):

        input = """

            main:function void()

            {

                a:string="aaa">="bbb">="ccc";

                return;

            }

        """

        expect = "Error on line 4 col 37: >="

        self.assertTrue(TestParser.test(input, expect, 290))

    def test\_91(self):

        input = """

            main:function void()

            {

                a:string="aaa">=("bbb">="ccc");

                return;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 291))

    def test\_92(self):

        input = """

            main:function void()

            {

                a:string="aaa"<=("bbb"<="ccc");

                return;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 292))

    def test\_93(self):

        input = """

            main:function void()

            {

                a:string="aaa"<="bbb">="ccc";

                return;

            }

        """

        expect = "Error on line 4 col 37: >="

        self.assertTrue(TestParser.test(input, expect, 293))

    def test\_94(self):

        input = """

            main:function void()

            {

                a:string="aaa"<"bbb">"ccc";

                return;

            }

        """

        expect = "Error on line 4 col 36: >"

        self.assertTrue(TestParser.test(input, expect, 294))

    def test\_95(self):

        input = """

           main:function void()

            {

                a:boolean=1==1!=1;

                return;

            }

        """

        expect = "Error on line 4 col 30: !="

        self.assertTrue(TestParser.test(input, expect, 295))

    def test\_96(self):

        input = """

           main:function boolean()

            {

                a:boolean=1==0&&0;

                return a;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 296))

    def test\_97(self):

        input = """

           main:function boolean()

            {

                a:integer = ---3;

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 297))

    def test\_98(self):

        input = """

           main:function boolean()

            {

                a:integer = -(2+3);

            }

        """

        expect = "successful"

        self.assertTrue(TestParser.test(input, expect, 298))

    def test\_99(self):

        input = """

           a,b: float = 1\_.2;

        """

        expect = "Error on line 2 col 25: \_"

        self.assertTrue(TestParser.test(input, expect, 299))

    def test\_100(self):

        input = """

           a,b: float = 1\_2.2;

        """

        expect = "Error on line 2 col 29: ;"

        self.assertTrue(TestParser.test(input, expect,300))