CS61AS Proj 4 - Schython

Part A: Eui Jin “Jeanie” Jung

Part B: Nguyet Minh Duong -- CS61AS-EU

**PERSON A**

**Common Exercise 2**

STk> (py-read)x = 3

(0 x = 3)

STk> (py-read) x = 3

(1 x = 3)

STk> (py-read) x = 3

(2 x = 3)

STk> (py-read) x = 3

(3 x = 3)

**Exercise A3**

STk> (py-read) print "Hello 'world'."

(1 print "Hello 'world'.")

STk> (py-read) print 'Hello 'world'.'

(1 print "Hello " world ".")

STk> (py-read) print 'Hello "world".'

(1 print "Hello \"world\".")

STk> (py-read) print 'hi my name is "blank"'

(1 print "hi my name is \"blank\"")

STk> (py-read) 'hi'. 123 "world"

(1 "hi" . 123 "world")

**Exercise A4**

STk> (define test-num-1 (make-py-num 3))

test-num-1

STk> (define test-num-2 (make-py-num 2))

test-num-2

STk> (define test-list

(make-py-list (list (make-py-num 3) (make-py-num 4)

(make-py-num 5) (make-py-num 6))))

test-list

STk> (ask (ask test-list '\_\_contains\_\_ test-num-1) 'true?)

#t

STk> (ask (ask test-list '\_\_contains\_\_ test-num-2) 'true?)

#f

STk> (ask test-list '\_\_contains\_\_ test-num-2)

#[closure arglist=(message) 52dd38]

STk> (ask (ask test-list '\_\_contains\_\_ (make-py-num 4)) 'true?)

#t

**Exercise A5**

>>> x = 3

>>> (x == 3)

True

>>> (x == 3) and (x == 3)

True

>>> (x == 3) and (x == 4)

False

>>> False and True

False

>>> True and 3 and 5

5

>>> True and 3 and False

False

>>> True or 3

True

>>> True or 3 or False

True

>>> False or True

True

>>> False

False

>>> False or False

False

>>> x = 3

>>> (x == 3) or (x == 4)

True

**Exercise A7**

>>> for i in [1, 2, 3, 4]:

... print i

...

1

2

3

4

>>> for i in [1, 3, 5, 2, 4]:

... print i

...

1

3

5

2

4

>>> for i in range(3, 10):

... print i

...

3

4

5

6

7

8

9

>>> for i in range(3, 10, 2):

... print i

...

3

5

7

9

>>> for i in range(7):

... print i

...

0

1

2

3

4

5

6

>>> for i in range(10, 13, 1):

... print i

...

10

11

12

13

**Common Exercise 8**

>>> x = {"hi" : "bye"}

>>> x

{

"hi" : "bye"

}

>>> x["hi"]

"bye"

>>> x = {"hello" : "world", "foo" : "bar"}

>>> x

{

"hello" : "world",

"foo" : "bar"

}

>>> x["foo"]

"bar"

>>> x["hello"]

"world"

>>> x = {"hello" : 5, "foo" : 6}

>>> x["hello"]

5

>>> x["foo"]

6

>>> x = {"hello" : 5, 1 : 3}

>>> x[1]

3

>>> x = {1 + 2 : 3 + 4}

>>> x

{

3 : 7

}

>>> x[1 + 2]

7

>>> x[3]

7

**PERSON B**

**Common Exercise 1**

(py-read)x = 3 # I am a comment IGNORE ME

(0 x = 3)

(py-read)x = 3 # om # nom # nom

(0 x = 3)

(py-read)x + 34 = 2 #4 - 2 HI THERE DOES THIS BREAK

(0 x + 34 = 2)

**Common Exercise 2**

(py-read) x\*4 = 4

(1 x \* 4 = 4)

(py-read) x + 3 = 4

(3 x + 3 = 4)

(py-read)x = 4

(0 x = 4)

**Exercise B3**

*;; Original Code*

STk> (py-read)x

(0 x)

STk> (py-read)4.5

(0 4 0.5)

*;; First Edit*

STk> (load "parser.scm")

okay

STk> (py-read)4.5

(0 4.5)

STk> (py-read)4.5.foo

(0 #f foo)

STk> (py-read)print 4.5

(0 print 4.5)

STk> (py-read)print 4.5 + 5.6

(0 print 4.5 + 5.6)

STk> (py-read)print 4.4.foobar

(0 print #f foobar)

*;; Second Edit*

STk> (load "parser.scm")

okay

STk> (py-read)print 4.4.foobar

(0 print #f foobar)

STk> (py-read)print 4.5 + 4.6

(0 print 4.5 + 4.6)

STk> (py-read)print 4.925 + 4.6.what

(0 print 4.925 + #f what)

*;; Third Edit*

STk> (load "parser.scm")

okay

STk> (py-read)print 4.5 + 5.6

(0 print 4.5 + 5.6)

STk> (py-read)print 4.925 + 4.6.what

(0 print 4.925 + okay .what)

STk> (py-read)print 4.4.foobar

(0 print okay .foobar)

*;; Fourth Edit*

STk> (load "parser.scm")

okay

STk> (py-read)print 4.4.foobar

(0 print 4.4 .foobar)

STk> (py-read)print 4.925 + 4.6.what

(0 print okay + 4.6 .what)

STk> (py-read)print 4.4.foobar

(0 print 4.4 .foobar)

*;; Fifth Edit -- PASSED*

STk> (load "parser.scm")

okay

STk> (py-read)print 4.4.foobar

(0 print 4.4 .foobar)

STk> (py-read)print 4.925 + 4.6.what

(0 print okay + 4.6 .what)

STk> (py-read)print 4.4.foobar

(0 print 4.4 .foobar)

**Exercise B4**

STk> (ask (negate-bool \*PY-TRUE\*) 'true?)

#f

STk> (ask (negate-bool \*PY-FALSE\*) 'true?)

#t

**Exercise B5 -- In**

>>> 1 in [1, 2, 3]

True

>>> 4 in [1, 2, 3]

False

**Exercise B5 -- Not**

>>> 1 not in [2, 3, 4]

\*\*\* Error at line 5 of file ./start.scm:

eval: bad function in : (object message)

*;; Second Edit -- PASSED*

>>> 1 not in [2, 3, 4]

True

>>> 1 not in [1, 2, 3]

False

**Common Exercise 6**

*;; First Edit*

>>> x = 2

>>> while x < 5:

... x = x + 1

...

sset

SyntaxError: bad block type: while-block\*

*;; Second Edit*

>>> x = 3

>>> while x < 5:

... print x

... x = x + 1

...

sdfdsf

NameError: Unbound variable: <

\*\*\* Error at line 5 of file ./start.scm:

PythonError

*;; Third Edit*

>>> x = True

>>> c = 4

>>> while x:

... print c

... x = False

...

k

\*\*\* Error at line 5 of file ./start.scm:

car: wrong type of argument: \*while-block\*

*;; Fourth Edit*

>>> x = 3

>>> while x < 6:

... print x

... x = x + 1

...

d

\*\*\* Error at line 5 of file ./start.scm:

cdr: wrong type of argument: \*block\*

*;; Fifth Edit*

>>> x

3

>>> x = 3

>>> while x < 5:

... print x

... x = x + 1

...

3

4

*;; Sixth(ish) Edit -- PASSED*

>>> x = 3

>>> while x < 5:

... print x

... x = x + 1

...

3

4

>>> x = 3

>>> while x < 5:

... print x

... break

...

3

>>> x = 3

>>> while x < 5:

... x = x + 1

... continue

...

>>> x

5

**Exercise B7**

*;; First Edit*

>>> x = 3

>>> if x == 4:

... print "foo"

... else:

... print "bar"

...

\*\*\* Error at line 5 of file ./start.scm:

eval: bad function in : (boolean-value)

*;; Second edit*

>>> x = 3

>>> if x == 4:

... print "foo"

... else:

... print "bar"

...

"bar"

>>> if x == 3:

... print x + 1

... else:

... if x < 4:

... print x + 2

... else:

... if x > 5:

... print x + 3

... else:

... print x + 4

...

4

*;; Third Edit*

>>> x = 5

>>> if x == 3:

... print "foo"

... elif x < 2:

... print "bar"

... elif x > 4:

... print "yessss"

... else:

... print "huh"

...

\*\*\* Error at line 5 of file ./start.scm:

set!: first argument is not a symbol: (cadr str-elif)

*;; Fourth Edit*

>>> x = 5

>>> if x == 2:

... print "foo"

... elif x == 3:

... print "bar"

... elif x == 4:

... print "ok"

... elif x == 5:

... print "yess"

... else:

... print "lol"

...

\*\*\* Error at line 5 of file ./start.scm:

car: wrong type of argument

*;; Fifth Edit -- Passed*

>>> x = 5

>>> if x == 3:

... print "foo"

... elif x == 4:

... print "bar"

... elif x == 5:

... print "yes"

... else:

... print "lol"

...

"yes"

**Common Exercise 8**

*;; First Edit*

>>> x = { 1 : "a", 2 : "b" }

SyntaxError: Expected comma separating key-value pairs in dictionary parsing

\*\*\* Error at line 5 of file ./start.scm:

PythonError

*;; Second Edit*

>>> x = { 1 : "a", 2 : "b" }

SyntaxError: Expected comma separating key-value pairs in dictionary parsing

\*\*\* Error at line 5 of file ./start.scm:

PythonError

*;; Third Edit*

>>> x = { 1 : "A", 2 : "B" }

\*\*\* Error at line 5 of file ./start.scm:

too few arguments to: (collect-key-value line-obj env)

*;; Fourth Edit*

STk> (load "start")

>>> x = { 1 : "A", 2 : "B" }

>>> x

{

1 : "A",

2 : "B"

}

>>> x = { "hello" : 1+5, "there" : 6+1, 4+5: 10 }

NameError: Unbound variable: +

\*\*\* Error at line 5 of file ./start.scm:

PythonError

*;; Fifth Edit -- Passed*

>>> x = { 1 : "A", 2 : "B" }

>>> x

{

1 : "A",

2 : "B"

}

>>> x = { "hello" : 1+5, "there" : 6+1, 4+5: 10 }

>>> x

{

"hello" : 6,

"there" : 7,

9 : 10

}

>>> x[9]

10

>>> x["there"]

7

>>> x[1+8]

10

**Common Exercise 9**

*;; First Edit*

>>> import memoize

>>> def fib(x, memo):

... if x <= 1:

... print x

... else:

... return memo(x-1) + memo(x-2)

...

>>> fib\_memo = memoize(fib)

>>> fib\_memo(1)

1

>>> fib\_memo(2)

0

AttributeError: objects of type <type 'NoneType'> have no method \_\_add\_\_

\*\*\* Error at line 5 of file ./start.scm:

PythonError

*;; Second Edit -- Passed*

>>> import memoize

>>> def f(x, memo):

... if x <= 1:

... return x

... else:

... return memo(x-1) + memo(x-2)

...

>>> fm = memoize(f)

>>> fm(1)

1

>>> fm(2)

1

>>> fm(3)

2

>>> fm(100)

354224848179261915075

>>> fm(20)

6765