Final Exam, Form 3 A CSE 231 Spring 2015

Name:____

Section:____________________ **Date:**____________________

INSTRUCTIONS:

(1) DO NOT OPEN YOUR EXAM BOOKLET UNTIL YOU HAVE BEEN TOLD TO BEGIN.

(2) Please fill in the requested information at the top of this exam booklet.

(3) Use a #2 pencil to encode answers on the OMR form (bubble sheet).

(4) Please encode the following on the OMR form:
– Last name and first initial
– MSU PID
– Section number (005, 006, 007, 008, 009, or 010)
– Exam form (3 A)

(5) Please sign the OMR form.

(6) Only answers recorded on your OMR form will be counted for credit. Completely erase any responses on the OMR form that you wish to delete.

(7) You may not ask questions once the examination has begun.

If there is a structural problem with your exam booklet (e.g., a missing or poorly printed page), please raise your hand; a proctor will take care of it.

If a question is ambiguous or contains a typographic error, write your interpretation of the question on the same page as the question; then put a note on the cover sheet of your exam booklet.

(8) *Choose the single best alternative for each question, even if you believe the question is ambiguous or contains an error.* If a question has more than one best answer, credit will be given for any of the correct answers provided that you marked only one answer.

(9) This exam booklet contains 40 questions, each of which will be weighted equally. The total points for the exam is 200 points (20% of your course grade).

(10) You may use one 8.5” x 11” note sheet and a paper dictionary during the exam. No other reference materials, calculating devices, or electronic devices may be used during the examination.

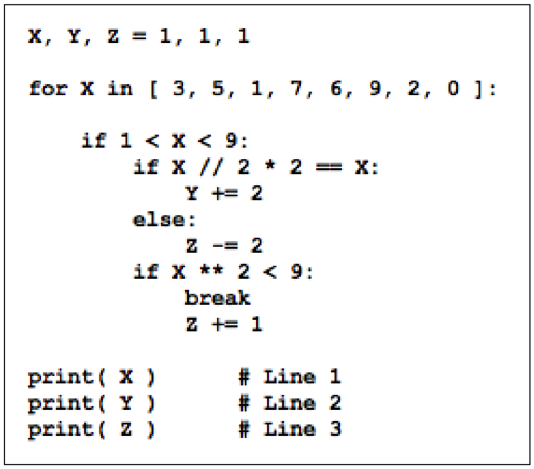
(11) The exam ends at 2:45 pm. You must turn in this exam booklet, the OMR form, your note sheet, and your scrap paper before leaving. Put your name on anything that you would like to have returned. When leaving, please be courteous to those still taking the exam.

```
Form A
  1   2   3   4   5   6   7   8   9  10
  B   D   C   C   C   D   B   C   C   B

 11  12  13  14  15  16  17  18  19  20
  E   D   D   D   A   B   C   B   C   C

 21  22  23  24  25  26  27  28  29  30
  B   C   C   C   A   A   A   A   B   B

 31  32  33  34  35  36  37  38  39  40
  D   B   B   C   C   E   A   C   C   B
```



```
X, Y, Z = 1, 1, 1

for X in [ 3, 5, 1, 7, 6, 9, 2, 0 ]:

    if 1 < X < 9:
        if X // 2 * 2 == X:
            Y += 2
        else:
            Z -= 2
        if X ** 2 < 9:
            break
            Z += 1

print( X )      # Line 1
print( Y )      # Line 2
print( Z )      # Line 3
```

Figure 1

(1) In Fig. 1, what is printed by the line labeled `Line 1`?

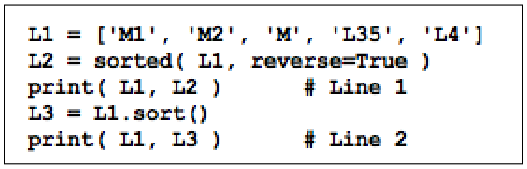
(a) `5` (b) `2` (c) `-5`
(d) `0` (e) None of (a)–(d)

(2) In Fig. 1, what is printed by the line labeled `Line 2`?

(a) `2` (b) `0` (c) `-4`
(d) `5` (e) None of (a)–(d)

(3) In Fig. 1, what is printed by the line labeled `Line 3`?

(a) `2` (b) `5` (c) `-5`
(d) `-4` (e) None of (a)–(d)



```
L1 = ['M1', 'M2', 'M', 'L35', 'L4']
L2 = sorted( L1, reverse=True )
print( L1, L2 )       # Line 1
L3 = L1.sort()
print( L1, L3 )       # Line 2
```

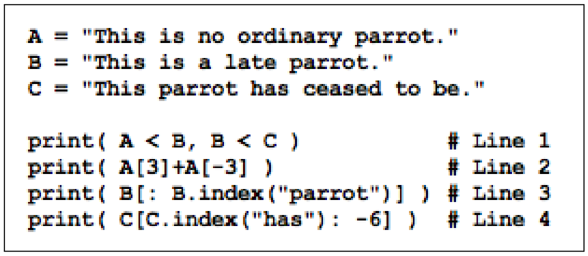
Figure 2

(4) In Fig. 2, what is printed by the line labeled `Line 1`?

(a) `['M1', 'M2', 'M', 'L35', 'L4'] ['L35', 'L4', 'M', 'M1', 'M2']`
(b) `['M1', 'M2', 'M', 'L35', 'L4'] None`
(c) `['M1', 'M2', 'M', 'L35', 'L4'] ['M2', 'M1', 'M', 'L4', 'L35']`
(d) `['M2', 'M1', 'M', 'L4', 'L35'] ['M2', 'M1', 'M', 'L4', 'L35']`
(e) None of (a)–(d)

(5) In Fig. 2, what is printed by the line labeled `Line 2`?

(a) `['M1', 'M2', 'M', 'L35', 'L4'] ['L35', 'L4', 'M', 'M1', 'M2']`
(b) `['M1', 'M2', 'M', 'L35', 'L4'] ['L4', 'L35', 'M', 'M1', 'M2']`
(c) `['L35', 'L4', 'M', 'M1', 'M2'] None`
(d) `['L35', 'L4', 'M', 'M1', 'M2'] ['L35', 'L4', 'M', 'M1', 'M2']`
(e) None of (a)–(d)



```
A = "This is no ordinary parrot."
B = "This is a late parrot."
C = "This parrot has ceased to be."

print( A < B, B < C )            # Line 1
print( A[3]+A[-3] )              # Line 2
print( B[: B.index("parrot")] )  # Line 3
print( C[C.index("has"): -6] )   # Line 4
```

FIGURE 3

(6) In Fig. 3, what is printed by the line labeled `Line 1`?

(a) `True True` (b) `False False`
(c) `True False` (d) `False True`
(e) None of (a)–(d).

(7) In Fig. 3, what is printed by the line labeled `Line 2`?

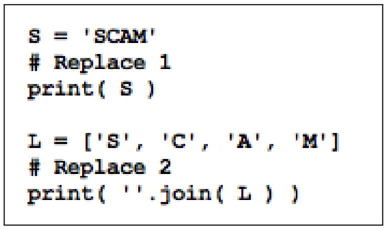
(a) `io` (b) `so`
(c) `sr` (d) `ir`
(e) None of (a)–(d).

(8) In Fig. 3, what is printed by the line labeled `Line 3`?

(a) `This is a late parrot`
(b) `This is a late p`
(c) `This is a late`
(d) An error
(e) None of (a)–(d)

(9) In Fig. 3, what is printed by the line labeled `Line 4`?

(a) `This parrot has`
(b) ` ceased`
(c) `has ceased`
(d) An error
(e) None of (a)–(d)



```
S = 'SCAM'
# Replace 1
print( S )

L = ['S', 'C', 'A', 'M']
# Replace 2
print( ''.join( L ) )
```

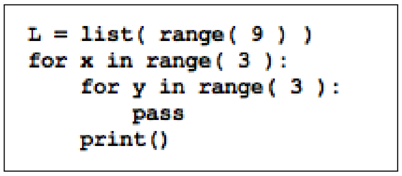
Figure 4

(10) In Fig. 4, which replacement for the comment `Replace 1` causes the first print statement to print `SLUM`?

(a) `S[1:3] = "LU"` (b) `S = S[:1] + 'LU' + S[3:]`
(c) `S[1], S[2] = 'L', 'U'` (d) None of (a)–(c).
(e) All of (a)–(c).

(11) In Fig. 4, which replacement for the comment `Replace 2` causes the second statement to print `SLUM`?

(a) `L[1:3] = "LU"` (b) `L = L[:1] + ['LU'] + L[3:]`
(c) `L[1], L[2] = 'L', 'U'` (d) None of (a)–(c).
(e) All of (a)–(c).



```
L = list( range( 9 ) )
for x in range( 3 ):
    for y in range( 3 ):
        pass
    print()
```

Figure 5

(12) In Fig. 5, which replacement for the `pass` statement causes the following to be printed:

```
0 3 6
1 4 7
2 5 8
```

(a) `print( L[-x - 3*y], end=" " )` (b) `print( L[y + 3*x], end=" " )`
(c) `print( L[x][3*x + y], end=" " )` (d) `print( L[x + 3*y], end=" " )`
(e) None of (a)–(d) is correct.

(13) In Fig. 5, which replacement for the `pass` statement causes the following to be printed:

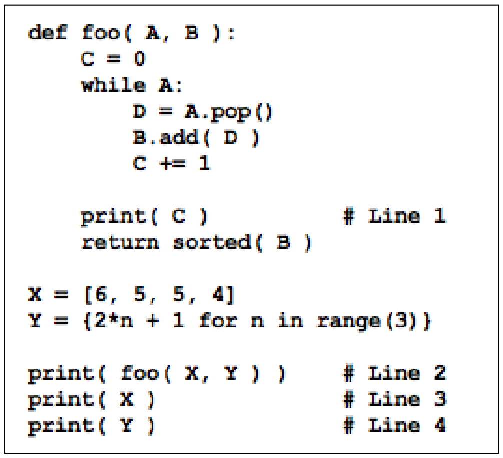
```
8 7 6
5 4 3
2 1 0
```

(a) `print( L[-x][-y], end=" " )` (b) `print( L[-3*y - x - 1], end=" " )`
(c) `print( L[-x][-3*x], end=" " )` (d) `print( L[-3*x - y - 1], end=" " )`
(e) None of (a)–(d) is correct.

(14) What is printed by the following code:

```
A = [1, 2]
B = A
C = ['A', B] + A
print( C )
```

(a) `[[1, 2], [1, 2], [1, 2]]` (b) `['A', [1, 2], [1, 2]]`
(c) `[1, 2, [1, 2], 1, 2]` (d) `['A', [1, 2], 1, 2]`
(e) None of (a)–(d) is correct.



```
def foo( A, B ):
    C = 0
    while A:
        D = A.pop()
        B.add( D )
        C += 1

    print( C )          # Line 1
    return sorted( B )

X = [6, 5, 5, 4]
Y = {2*n + 1 for n in range(3)}

print( foo( X, Y ) )    # Line 2
print( X )              # Line 3
print( Y )              # Line 4
```

Figure 6

(15) In Fig. 6, what is printed by the line labeled `Line 1`?

(a) `4` (b) `3` (c) `1`
(d) `0` (e) None of (a)–(d)

(16) In Fig. 6, what is printed by the line labeled `Line 2`?

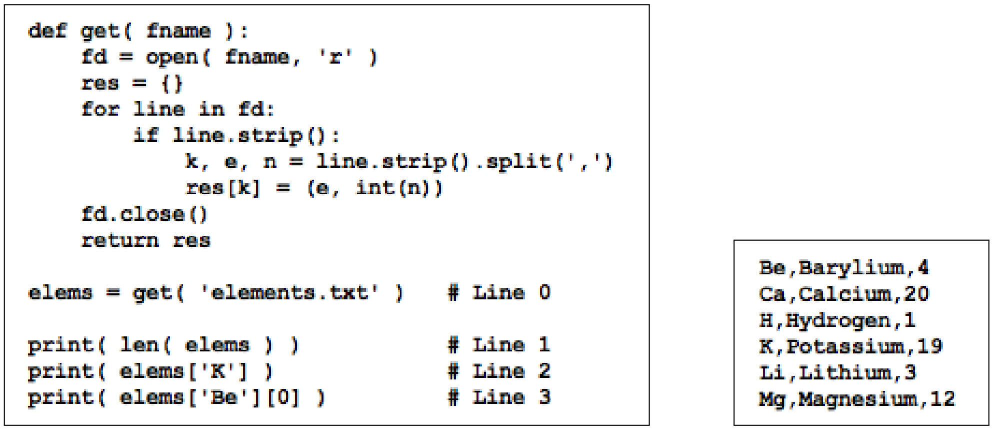
(a) `[6, 5, 4]` (b) `[1, 3, 4, 5, 6]` (c) `[]`
(d) `{1, 3, 4, 5, 6}` (e) None of (a)–(d)

(17) In Fig. 6, what is printed by the line labeled `Line 3`?

(a) `[1, 3, 4, 5, 6]` (b) `[6, 5, 5, 4]` (c) `[]`
(d) `{1, 3, 4, 5, 6}` (e) None of (a)–(d)

(18) In Fig. 6, what is printed by the line labeled `Line 4`?

(a) `[1, 3, 4, 5, 6]` (b) `{1, 3, 4, 5, 6}` (c) `[]`
(d) `{1, 3, 5}` (e) None of (a)–(d)



```
def get( fname ):
    fd = open( fname, 'r' )
    res = {}
    for line in fd:
        if line.strip():
            k, e, n = line.strip().split(',')
            res[k] = (e, int(n))
    fd.close()
    return res

elems = get( 'elements.txt' )    # Line 0

print( len( elems ) )            # Line 1
print( elems['K'] )              # Line 2
print( elems['Be'][0] )          # Line 3
```

```
Be,Barylium,4
Ca,Calcium,20
H,Hydrogen,1
K,Potassium,19
Li,Lithium,3
Mg,Magnesium,12
```

FIGURE 7

(19) Which of the following statements about the call to `get` in `Line 0` of Fig. 7 is correct?
(a) It always returns a dictionary.
(b) It will produce an error if a file named `elements.txt` exists but is empty.
(c) It will produce an error if a file named `elements.txt` does not exist.
(d) None of (a)–(c) is correct. (e) All of (a)–(c) are correct.

(20) If the contents of file `elements.txt` are as shown on the bottom right of Fig. 7, what is printed by `Line 1`?

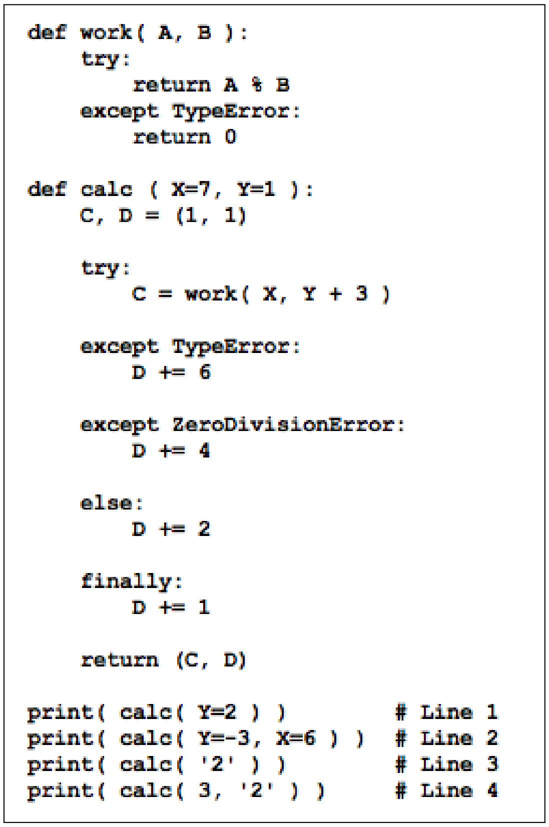
(a) `0` (b) `18` (c) `6`
(d) `3` (e) None of (a)–(d)

(21) If the contents of file `elements.txt` are as shown on the bottom right of Fig. 7, what is printed by `Line 2`?

(a) `Potassium` (b) `('Potassium', 19)` (c) `19`
(d) An error (e) None of (a)–(d)

(22) If the contents of file `elements.txt` are as shown on the bottom right of Fig. 7, what is printed by `Line 3`?

(a) `(Barylium, 4)` (b) `4` (c) `Barylium`
(d) An error (e) None of (a)–(d)



```
def work( A, B ):
    try:
        return A % B
    except TypeError:
        return 0

def calc ( X=7, Y=1 ):
    C, D = (1, 1)

    try:
        C = work( X, Y + 3 )

    except TypeError:
        D += 6

    except ZeroDivisionError:
        D += 4

    else:
        D += 2

    finally:
        D += 1

    return (C, D)

print( calc( Y=2 ) )          # Line 1
print( calc( Y=-3, X=6 ) )   # Line 2
print( calc( '2' ) )          # Line 3
print( calc( 3, '2' ) )      # Line 4
```

Figure 8

(23) In Fig. 8, what is printed by the line labeled `Line 1`?

(a) `(2, 2)` (b) `(0, 3)` (c) `(2, 4)`
(d) `(0, 2)` (e) None of (a)–(d)

(24) In Fig. 8, what is printed by the line labeled `Line 2`?

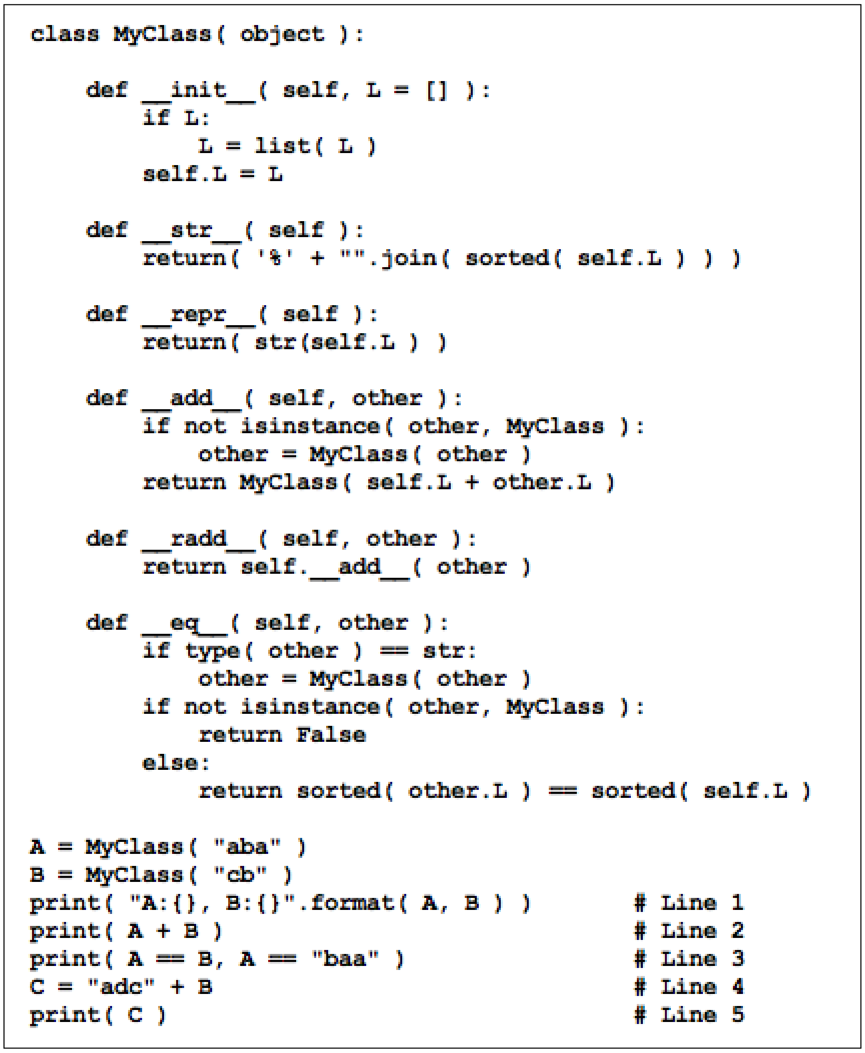
(a) `(0, 6)` (b) `(6, 8)` (c) `(1, 6)`
(d) `(1, 8)` (e) None of (a)–(d)

(25) In Fig. 8, what is printed by the line labeled `Line 3`?

(a) `(0, 4)` (b) `(0, 8)` (c) `(0, 3)`
(d) `(1, 8)` (e) None of (a)–(d)

(26) In Fig. 8, what is printed by the line labeled `Line 4`?

(a) `(1, 8)` (b) `(0, 8)` (c) `(1, 7)`
(d) `(0, 4)` (e) None of (a)–(d)



```
class MyClass( object ):

    def __init__( self, L = [] ):
        if L:
            L = list( L )
        self.L = L

    def __str__( self ):
        return( '%' + "".join( sorted( self.L ) ) )

    def __repr__( self ):
        return( str(self.L ) )

    def __add__( self, other ):
        if not isinstance( other, MyClass ):
            other = MyClass( other )
        return MyClass( self.L + other.L )

    def __radd__( self, other ):
        return self.__add__( other )

    def __eq__( self, other ):
        if type( other ) == str:
            other = MyClass( other )
        if not isinstance( other, MyClass ):
            return False
        else:
            return sorted( other.L ) == sorted( self.L )

A = MyClass( "aba" )
B = MyClass( "cb" )
print( "A:{}, B:{}".format( A, B ) )        # Line 1
print( A + B )                              # Line 2
print( A == B, A == "baa" )                 # Line 3
C = "adc" + B                               # Line 4
print( C )                                  # Line 5
```

Figure 9

(27) In Fig. 9, the `MyClass` string conversion method (`__str__`) is *not* called during execution of which line?

(a) `Line 4` (b) `Line 2` (c) `Line 1`
(d) `Line 5` (e) It is called for all.

(28) In executing `Line 2` of Fig. 9, the shell maps the expression `A + B` to which of the following expressions?

(a) `A.__add__(B)` (b) `self.__add__(B)` (c) `B.__radd__(A)`
(d) `sum( A, B )` (e) None of (a)–(d).

(29) In executing `Line 3` of Fig. 9, the shell maps the expression `A == B` to which of the following expressions?

(a) `self.__eq__(B)` (b) `A.__eq__(B)` (c) `B.__eq__(A)`
(d) `isequal( A, B )` (e) None of (a)–(d).

(30) When Fig. 9 is executed, what is displayed by `Line 1`?

(a) `A:%aba, B:%cb`
(b) `A:%aab, B:%bc`
(c) `A:[’a’, ’a’, ’b’] B:[’c’, ’b’]`
(d) `A:[’a’, ’b’, ’a’] B:[’c’, ’b’]`
(e) None of (a)–(d).

(31) When Fig. 9 is executed, what is displayed by `Line 2`?

(a) `[’a’, ’b’, ’a’, ’c’, ’b’]`
(b) `%ababc`
(c) `[’a’, ’a’, ’b’, ’c’, ’b’]`
(d) `%aabbc`
(e) None of (a)–(d).

(32) When Fig. 9 is executed, what is displayed by `Line 3`?

(a) `False False`
(b) `False True`
(c) `True False`
(d) `True True`
(e) None of (a)–(d).

(33) When Fig. 9 is executed, what is displayed by `Line 5`?

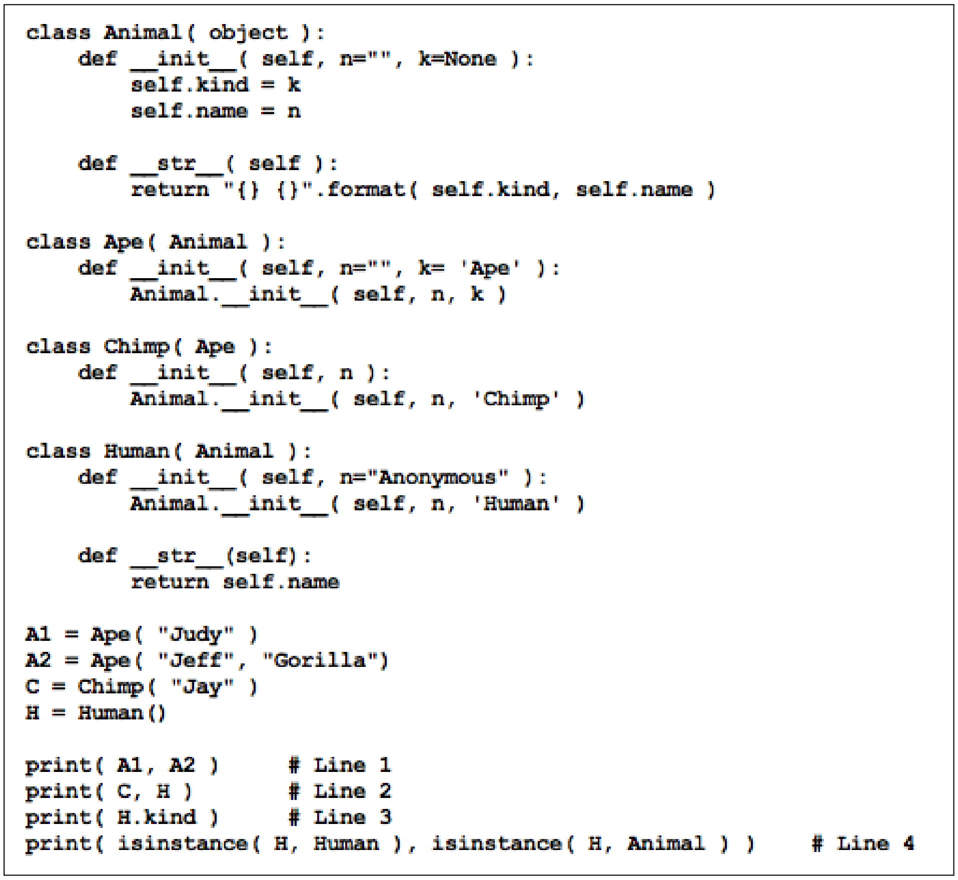
(a) `%cbadc`
(b) `%abccd`
(c) `%adccb`
(d) `[’a’, ’b’, ’a’, ’c’, ’b’]`
(e) None of (a)–(d).

(34) After executing Fig. 9, what does the Python shell display if you type `A` at the shell prompt (`>>>`) and then press return?

(a) `%aba`
(b) `%aab`
(c) `[’a’, ’b’, ’a’]`
(d) `[’a’, ’a’, ’b’]`
(e) None of (a)–(d).

(35) After executing Fig. 9, what does the Python shell display if you type `C` at the shell prompt (`>>>`) and then press return?

(a) `[’a’, ’b’, ’c’, ’d’]`
(b) `[’a’, ’b’, ’c’, ’c’, ’d’]`
(c) `[’c’, ’b’, ’a’, ’d’, ’c’]`
(d) `[’a’, ’d’, ’c’, ’c’, ’b’]`
(e) None of (a)–(d).



```
class Animal( object ):
    def __init__( self, n="", k=None ):
        self.kind = k
        self.name = n

    def __str__( self ):
        return "{} {}".format( self.kind, self.name )

class Ape( Animal ):
    def __init__( self, n="", k= 'Ape' ):
        Animal.__init__( self, n, k )

class Chimp( Ape ):
    def __init__( self, n ):
        Animal.__init__( self, n, 'Chimp' )

class Human( Animal ):
    def __init__( self, n="Anonymous" ):
        Animal.__init__( self, n, 'Human' )

    def __str__(self):
        return self.name

A1 = Ape( "Judy" )
A2 = Ape( "Jeff", "Gorilla")
C = Chimp( "Jay" )
H = Human()

print( A1, A2 )     # Line 1
print( C, H )       # Line 2
print( H.kind )     # Line 3
print( isinstance( H, Human ), isinstance( H, Animal ) )    # Line 4
```

FIGURE 10

(36) Which of the following is present in Fig. 10?

(a) inheritance (b) subtypes (c) supertypes
(d) None of (a)–(c) (e) All of (a)–(c)

(37) In Fig. 10, what is printed by the line labeled `Line 1`?

(a) `Ape Judy Gorilla Jeff` (b) `Human Judy Human Jeff`
(c) `Judy Jeff` (d) An error
(e) None of (a)–(d)

(38) In Fig. 10, what is printed by the line labeled `Line 2`?

(a) `Chimp Jay` (b) `Human Jay Human`
(c) `Chimp Jay Anonymous` (d) An error
(e) None of (a)–(d)

(39) In Fig. 10, what is printed by the line labeled `Line 3`?

(a) `Anonymous` (b) `Animal` (c) `Human`
(d) An error (e) None of (a)–(d)

(40) In Fig. 10, what is printed by the line labeled `Line 4`?

(a) `True False` (b) `True True` (c) `False False`
(d) An error (e) None of (a)–(d)

Scratch