Intro to Python
Module 1
Exam 1
October 21, 2017
Time Limit: 80 Minutes

Name: ______________________________

Teaching Assistant ______________________________

This exam contains 10 pages (including this cover page) and 33 questions. Total of points is 40.

Question:	1	2	3	4	5	6	7	8	9	10	11	Total
Points:												
Score:												

Question:	12	13	14	15	16	17	18	19	20	21	22	Total
Points:												
Score:												

Question:	23	24	25	26	27	28	29	30	31	32	33	Total
Points:												
Score:												

1. (2 points) Consider the generator `foo` is

```
def foo(**kwargs):
    yield from kwargs
```

   Mark box if **true**.

   ◯
   ```
   >>> for i in foo(1, 2, 3): print(i, end=' ')
   1 2 3
   ```
   ◯
   ```
   >>> for i in foo(a=1, b=2, c=3): print(i, end=' ')
   a c b
   ```
   ◯
   ```
   >>> for i in foo(a=1, c=2, c=3): print(i, end=' ')
   {'a': 1, 'c': 3, 'b': 3}
   ```
   ◯
   ```
   >>> for i in foo(a=1, b=2, c=3): print(i, end=' ')
   1 2 3
   ```
   ◯
   ```
   >>> for i in foo(1, 2, 3): print(i, end=' ')
   (1, 2, 3)
   ```

2. (2 points) Mark box if the expression is **false**.

   ◯ `False == False in [False]`
   ◯ `[1, 2, 3] == sorted([3, 2, 1])`
   ◯ `[1, 2, 3] == [3, 2, 1].sort()`
   ◯ `min({1: 2, -1: -2}, key={1: 2, -1: -2}.get) == -1`
   ◯ `max({1: 2, -1: -2}, key={1: 2, -1: -2}.get) == 2`

3. (1 point) Consider list is `array = [1, 2, 3]`. What is the following method removes **last element** from `array`?

   A. `array.remove(-1)`
   B. `array.index(2)`
   C. `del array[array.index(2)]`
   D. `array.pop(-1)`
   E. `array.pop(array[-1])`

4. (1 point) Consider that `what = lambda arg: set(dir(arg))`. What is the value of expression `(what([]) & what("") & what({}) - what(0))`?

   A. CPython raises exception with type `SyntaxError`
   B. `{'__cls__', '__init__', '__repr__', '__str__'}`
   C. `{'__contains__', '__getitem__', '__iter__', '__len__'}`
   D. `{'__contains__', '__getitem__', '__iter__', 'index'}`
   E. `{'__contains__', '__getitem__', '__iter__'}`

5. (2 points) Which data types are an example of hash table?

   ◯ `list`
   ◯ `dict`
   ◯ `set`
   ◯ `deque`
   ◯ `array`

6. (1 point) Consider that

```
class Base: pass
class A(Base): pass
class B(Base): pass
class C: pass
class D(A, B, C): pass
```

The `D.__mro__` is equal to

A. `(__main__.D, __main__.A, __main__.B, __main__.Base, __main__.C, object)`

B. `(__main__.A, __main__.B, __main__.Base, __main__.C, object)`

C. `(__main__.D, __main__.A, __main__.B, __main__.Base, __main__.C)`

D. `(__main__.D, __main__.A, __main__.B, __main__.C, __main__.Base, object)`

E. `(__main__.D, __main__.C, __main__.A, __main__.B, __main__.Base, object)`

7. (1 point) What is the output of the following code?

```
print(type(lambda: None))
```

A. CPython raises exception with type `SyntaxError`.

B. `<class 'NoneType'>`

C. `<class 'type'>`

D. `<class 'tuple'>`

E. `<class 'function'>`

8. (1 point) What gets printed?

```
import re
sum_ = 0
pattern = 'back'
if re.match(pattern, 'backup.txt'):
    sum_ += 1
if re.match(pattern, 'text.back'):
    sum_ += 2
if re.search(pattern, 'backup.txt'):
    sum_ += 4
if re.search(pattern, 'text.back'):
    sum_ += 8
print(sum_)
```

A. `3`

B. `7`

C. `13`

D. `14`

E. `15`

9. (1 point) Why instance of class `list` can't be used as dictionary keys ?

A. Because lists are immutable and therefore not hashable.

B. Because lists are mutable and therefore not hashable.

C. Because lists can have duplicate elements.

D. Because lists can have unhashable elements.

E. Lists CAN be used as dictionary keys.

10. (1 point) What is the output of the following code?

```
arg = 1
def foo(arg):
    arg = 2
    def bar():
        bar.arg = arg
        return bar.arg
    return bar
bar = foo(arg)
print(bar(), bar.arg)
```

   A. CPython raises exception with type `SyntaxError`.

   B. CPython raises exception with type `AttributeError`.

   C. `1 1`

   D. `2 2`

   E. `<class 'function'> 2`

11. (2 points) What is the output of the following code?

```
arg = [1]
def foo(arg=2):
    arg.append(arg)
    def bar():
        return bar.arg
    bar.arg = arg
    return bar
bar = foo()
print(bar(), bar.arg)
```

   A. CPython raises exception with type `SyntaxError`.

   B. CPython raises exception with type `AttributeError`.

   C. `[1] [1, 2]`

   D. `[1] [1, [1]]`

   E. `[1, [...]] [1, [...]]`

12. (2 points) Which of the following is true about generators?

   ◯ Generators must contain a `yield` statement.

   ◯ Generator have a `__next__` method.

   ◯ Generator are iterators which create their elements on-the-fly.

   ◯ Generators should not contain a `return` statement.

   ◯ Generators have a `__getitem__` method.

13. (1 point) Which data type is an example of binary search tree?

   A. `Counter`

   B. `queque`

   C. `ChainMap`

   D. `OrderedDict`

   E. None of the above

14. (1 point) What is the output of the following code?

```
l = [1, 2, 3, 4, 5, 6]
def gen():
    it = iter(l)
    next(it)
    yield from it
for i in gen(): print(i, end=' ')
```

A. CPython raises exception of type `NameError`.

B. CPython raises exception of type `TypeError`.

C. `1 2 3 4 5 6`

D. `2 4 6`

E. `2 3 4 5 6`

15. (1 point) What is the output of the following code?

```
class Container:
    data = []
class List(Container):
    def append(self, value):
        self.data.append(value)
l = List()
l.append(1)
l.append(2)
print(Container.data)
```

A. CPython raises exception of type `NameError`

B. `[]`

C. `[1, 2]`

D. CPython raises exception of type `AttributeError`

E. CPython raises exception of type `RecursionError`

16. (1 point) What is the output of the following code?

```
class Foo:
    def foo(self = []):
        print(self, end=' ')
    def __str__(self):
        return 'foo'
print(Foo().foo(), Foo.foo(), end=' ')
```

A. `[] foo None None`

B. CPython raises exception of type `TypeError`

C. `None None foo foo`

D. `foo [] None None`

E. `[] [] None None`

17. (1 point) What are the time and space complexities of the `list.sort()` method?

A. Time complexity: $O(n)$. Space complexity: $O(1)$

B. Time complexity: $O(n \log n)$. Space complexity: $O(1)$

C. Time complexity: $O(n)$. Space complexity: $O(n)$

D. Time complexity: $O(n \log n)$. Space complexity: $O(n)$

E. Time complexity: $O(n \log n)$. Space complexity: $O(n \log n)$

18. (1 point) What is the output of the following code?

```
a = [1]
b = a
b.append(a)
print(a is b, b == a, a is b[1], b is a[1])
```

   A. `True True True True`

   B. `True True False False`

   C. `True False False False`

   D. CPython raises exception of type `RecursionError`

   E. CPython raises exception of type `IndexError`

19. (1 point) What are the time and space complexities of the `for i in range(n): i ** 2`?

   A. Time complexity: $O(n)$. Space complexity: $O(1)$

   B. Time complexity: $O(n^2)$. Space complexity: $O(n)$

   C. Time complexity: $O(1)$. Space complexity: $O(1)$

   D. Time complexity: $O(n)$. Space complexity: $O(n)$

   E. Time complexity: $O(n^2)$. Space complexity: $O(1)$

20. (1 point) What is the output of the following code?

```
class Context:
    data = []
    def __enter__(self):
        return
    def __exit__(self, etype, eref, etb):
        1/0
        return True
with Context() as c:
    c.data.append(1)
    print(c.data)
```

   A. `[1]`

   B. CPython raises exception of type `TypeError`

   C. CPython raises exception of type `AttributeError`

   D. Cpython raises exception of type `ZeroDivisionError`

   E. `[]`

21. (1 point) What is the output of the following code?

```
class Foo:
    foo = []
    def foo(self):
        Foo.foo.append('foo')
Foo().foo()
print(Foo.foo)
```

   A. `['foo']`

   B. CPython raises exception of type `TypeError`

   C. CPython raises exception of type `AttributeError`

   D. Cpython raises exception of type `RecursionError`

   E. `[]`

22. (1 point) What is the output of the following code?

```
def foo():
    foo.x = 'foo'
    return foo
print(foo.__call__().x)
```

A. `foo`

B. `AttributeError: 'function' object has no attribute 'x'`

C. `AttributeError: 'function' object has no attribute '__call__'`

D. `None`

E. `<function __main__.foo>`

23. (1 point) What is the output of the following code?

```
class Context:
    data = []
    def __enter__(self):
        return self
    def __exit__(self, etype, eref, etb):
        return True
with Context() as c:
    c.data.append(1)
    print(c.data)
```

A. `[1]`

B. CPython raises exception of type `TypeError`

C. CPython raises exception of type `AttributeError`

D.

E. `[]`

24. (1 point) What is the output of the following code?

```
def foo(*args, **kwargs):
    return args, kwargs
a, *b, c = foo(1, x=2)
print(a, b, c)
```

A. `(1,) [] 2`

B. CPython raises exception of type `TypeError`

C. CPython raises exception of type `SyntaxError`

D. `(1,) [] {'x': 2}`

E. `(1,) [] x`

25. (2 points) Which data types are immutable?

◯ `str`

◯ `dict`

◯ `set`

◯ `tuple`

◯ `ordered_map`

26. (1 point) What is the output of the following code?

```
class Context:
    data = []
    def __enter__(self):
        return
    def __exit__(self, etype, eref, etb):
        return True
with Context() as c:
    c.data.append(1)
    print(c.data)
```

A. `[1]`

B. CPython raises exception of type `TypeError`

C. CPython raises exception of type `AttributeError`

D.

E. `[]`

27. (1 point) What is the output of the following code?

```
def foo():
    foo = 'foo'
    print(foo, end=' ')
    def bar():
        nonlocal foo
        foo = 'bar'
    print(foo, end=' ')
    return bar
foo()()
```

A. `foo bar`

B. CPython raises exception of type `TypeError`

C. CPython raises exception of type `SyntaxError`

D. `bar bar`

E. `foo foo`

28. (1 point) What is the output of the following code?

```
def foo():
    foo = 'foo'
    print(foo, end=' ')
    def bar():
        nonlocal foo
        foo = 'bar'
        print(foo, end=' ')
    return bar
foo()()
```

A. `foo bar`

B. CPython raises exception of type `TypeError`

C. CPython raises exception of type `SyntaxError`

D. `bar bar`

E. `foo foo`

29. (1 point) What are the time and space complexities of the following procedure?

```python
def minimum(array): return sorted(array)[0]
```

A. Time complexity: $O(n)$. Space complexity: $O(1)$
B. Time complexity: $O(n \log n)$. Space complexity: $O(1)$
C. Time complexity: $O(n)$. Space complexity: $O(n)$
D. Time complexity: $O(n \log n)$. Space complexity: $O(n)$
E. Time complexity: $O(n \log n)$. Space complexity: $O(n \log n)$

```python
class Foo:
    foo = 'bar'
    def __getattr__(self, attr):
        if attr == 'foo':
            return 'foo'
        if attr == 'bar':
            raise AttributeError('I\'m bot bar!')
        return super().__getattribute__(attr)
    def __setattr__(self, attr, value):
        if attr == 'foo':
            setattr(Foo, attr, value)
        super().__setattr__(attr, value)
```

30. (2 points) Mark box if the expression is **true**.

- ◯ `'foo' in Foo.__dict__`
- ◯ `'foo' in Foo().__dict__`
- ◯ `'bar' in Foo.__dict__`
- ◯ `hasattr(Foo(), 'foo')`
- ◯ `hasattr(Foo, 'bar')`

31. (1 point) What is the output of the following code?

```python
print(Foo('bar').foo)
```

A. CPython raises exception of type `AttributeError`.
B. CPython raises exception of type `TypeError`.
C. `foo`
D. `I'm bot bar!`
E. `bar`

32. (1 point) What is the output of the following code?

```python
foo = Foo()
foo.bar = 'foo'
foo.foo = 'bar'
Foo.foo = 'foo'
print(foo.bar, Foo.foo, foo.foo)
```

A. `foo bar foo`
B. CPython raises exception of type `AttributeError`.
C. `bar foo foo`
D. `foo foo foo`
E. `foo foo bar`

33. (1 point) Who is Guido van Rossum?

    A. Creator of C++

    B. The president of Python

    C. Rapping artist

    D. Creator of package `this`

    E. Benevolent Dictator For Life