Internal Assessment - 1

Subject : Introduction to Programming with Python
Maximum Marks : 30 **Time: 2 hrs.**

Please read the following instructions carefully.

- You are required to draw vertical lines in your code (wherever needed) to clearly show indentation. For instance:

```
def foo():
    if something:
        do something
        do more things
    do something last
```

- Write a precise explanation wherever asked for.

1. What is the output of the following fragments of code? Justify your answer. 12 points

(i)
```
def myfunc(n):
  n = 2*n
  return n

myfunc(5)
print(n)
```

NAME ERROR

(ii)
```
y = 10
x = y == 10
print(x)
x = y = 15
print(x)
print(y)
```

TRUE
15
15

(iii) The following commands are given sequentially at the python shell.

```
>>> import myfile
>>> myfile.demo()
```

The contents of the file **myfile.py** are given below.

```
def demo():
        print("I am here")

print("I am there too")
```

Mention and explain the output you would expect.

I am there too
I am here

(iv)
```
var1 = 1
var2 = 2
var3 = "3"
print(var1+var2+var3)
```

TypeError: unsupported operand types for +

2. Write a single line of code for each of the following: 6 points

(i) A certain function is supposed to receive a float number and do some processing on it. The function signature is as follows: **fun(x)**
Write a single line of code to receive a float number from the user, round it to 2 decimal places and use it as an argument to make a call

to the above function. Assume that the user knows that he has to input a float number, so no prompt message is needed. The command **help(round)** in python flashes the following text on the screen:

```
round(number, ndigits=None)
    Round a number to a given precision in decimal
    digits.

    The return value is an integer if ndigits is
    omitted or None. Otherwise the return value has
    the same type as the number. ndigits may be
    negative.
```

fun(round(float(input()),2))

(ii) The number formed by dropping the last digit of a number **num** is explained through the following examples:

- The number formed by dropping the last digit of 1035 is 103
- The number formed by dropping the last digit of 200 is 20
- The number formed by dropping the last digit of 56 is 5
- The number formed by dropping the last digit of 4 is 0

A function having three local variables **x**, **y** and **z** is supposed to return true if either of the following two conditions is satisfied:

- **x** is more than **y-x**, 2^x is greater than the number formed by dropping the last digit of **y**, and **z** is equal to 30
- **x** is no more than **y-x**

Encode the above in the return statement for the function. Make sure to design the expression so as to avoid any redundant (extra) work.

return x<=y-x or z==30 and 2**x>y//10

3. A student wrote the following code to increment the value of variable x by 10. 3 points

x = 25

```
def myfunc():
  x = x + 10

myfunc()
print(x)
```

Explain what is wrong with the code. Modify to make it work while ensuring that you **DO NOT** i) use the 'global' keyword (ii) change the signature of the function. **Hint:** think about i) fruitful functions ii) modifying x within myfunc()

```
x = 25
def myfunc():
   t = x + 10
   return t

x = myfunc()
print(x)
```

4. A function receives the following two parameters : i) number of chocolates ii) total number of children, and returns the number of chocolates that each child gets. The chocolates cannot be divided, so no child gets half of a chocolate. In case all the chocolates cannot be evenly divided amongst the students, the remaining ones are just left. The header of the "yet to be implemented function" is given below. Mention the data types of the input parameters and the return value. Write at least 3 input-output examples (as discussed in class) that you think are "special cases" and will be instrumental in understanding the working of the function. 4 points

```
def divide_choc(choc_num,child_num):

choc_num: integer
child_num: integer
choc_per_child: integer
>>> divide_choc(18,4)
```

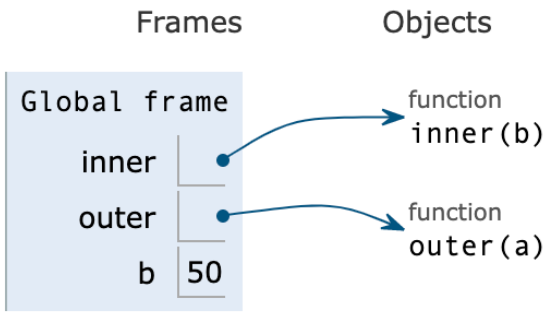
```
4
>>> divide_choc(0,3)
0
>>> divide_choc(10,0)
-1
>>> divide_choc(-10,2)
-1
>>> divide_choc(10,-2)
-1
```

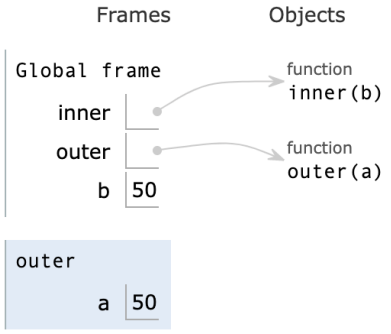
5. 5 points

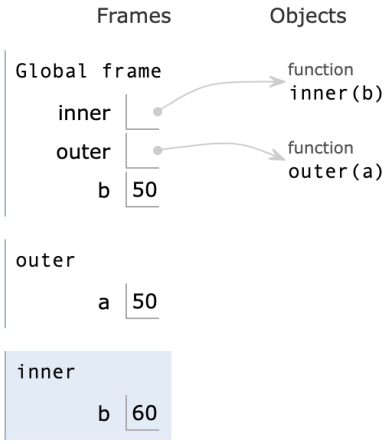
```
def inner(b):
  a = 2
  return b/a

def outer(a):
  b = inner(a+10)
  return b
```

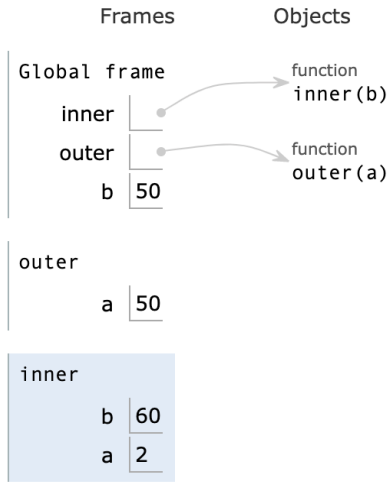
Draw all call frames for the following (start after the given call frame):

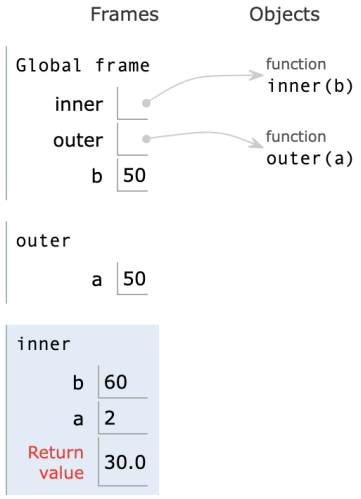
```
b = 50
a = outer(b)
```

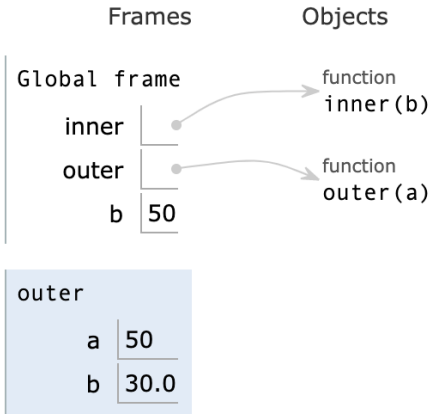


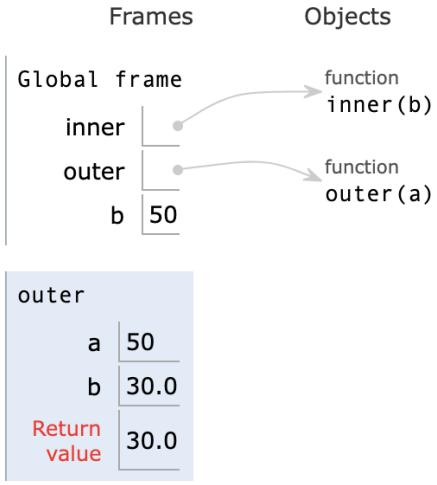

Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50

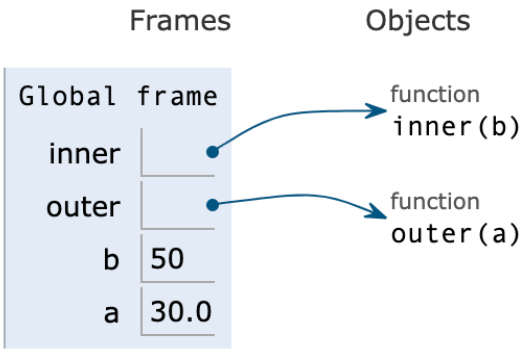

Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50
inner
b
60


Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50
inner
b
60
a
2


Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50
inner
b
60
a
2
Return value
30.0


Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50
b
30.0


Frames
Objects
Global frame
inner
outer
b
50
function
inner(b)
function
outer(a)
outer
a
50
b
30.0
Return value
30.0


Frames
Objects
Global frame
inner
outer
b
50
a
30.0
function
inner(b)
function
outer(a)