Python Function Practice

```
== Function Practices==================================


1. Write a function f1(list) that will return the number of odd
elements in a given list.

>>> f1([1,2,3,4])
2
>>> f1([1,2,3,4,5])
3




2. Write a function f2(list) that will print each odd element in a
given list.

>>> f2([1,2,3,4])
1
3
>>> f2([1,2,3,4,5])
1
3
5
```

3. Write a function f3(list) that will return the sum of all odd elements in a given list.

```
>>> f3([1,2,3,4])
4
>>> f3([1,2,3,4,5])
9
```

4. Write a function f4(list) that will return the sum of all the index positions whose corresponding element is odd in a given list.

```
>>> f4([1,2,3,4])
2
>>> f4([1,2,3,4,5])
6
```

5. Write a function f5(list) that will return the same list where each element has been squared.

```
>>> f5([1,2,3,4])
[1, 4, 9, 16]
>>> f4([1,2,3,4,5])
[1, 4, 9, 16, 25]
```

6. Write a function f6(list) that will return the largest number in a given list

```
>>> f6([1,2,3,4])
4
>>> f6([1,2,3,4,5])
5
```

7. Write a function f7(list) that will return the average of all the numbers in a given list.

```
>>> f7([1,2,3,4])
2.5
>>> f7([1,2,3,4,5])
3.0
```

8. Write a function f8(a,b,n) that will print all the numbers divisible by |n| within the range |a| and |b| inclusive. Assume |n| is positive.

```
>>> f8(1,10,2)
2
4
6
8
10
>>> f8(1,10,11)
>>> f8(1,10,7)
7
```

9. Write a function f9(width,height) that will print an ASCII rectangle with the given width and height.

```
>>> f9(0,1)
>>> f9(10,0)
>>> f9(1,1)
*
>>> f9(1,2)
*
*
>>> f9(5,5)
*****
*****
*****
*****
*****
```

10. Write a function f10(n) that will print a triangle with the given height |n|. Assume |n| is nonnegative.

```
>>> f10(1)
*
>>> f10(2)
*
**
>>> f10(3)
*
**
***
```

11. Write a function f11(list) that will return |True| if the list is sorted in *descending* order and |False| otherwise. Return |True| for the empty list.

```
>>> f11([])
True
>>> f11([5,4,3,2,1])
True
>>> f11([5,4,3,2,0])
True
>>> f11([5,4,5,2])
False
```

12. Write a function f12(list) that will return |True| if the list consists of all negative numbers and |False| otherwise. Return |True| for the empty list.

```
>>> f12([])
True
>>> f12([-1,-2,-3,-4,5])
False
>>> f12([1,2,3,4,5])
False
>>> f12([-1,-2,-3])
True
```

13. Write a function f13(list,target) that will return the index of the last occurrence of target in the list. Assume the list is nonempty and always contains the target.

```
>>> f13([1,2,3], 3)
2
>>> f13([1,2,3,1,2,3], 3)
5
>>> f13([1,1,1,1], 1)
3
```

14. Write a function f14(list) that will return the index of the last negative number in the list. Assume the list is nonempty and always contains a negative number.

```
>>> f14([1,2,-3])
2
>>> f14([1,-2,-3,1,-2,-3])
5
>>> f14([-1,1,1,1])
0
```

15. Write a function f15(list) that will return the sum of all the elements at even index positions.

```
>>> f15([1,2,-3])
-2
>>> f15([1,-2,-3,1,-2,-3])
-4
>>> f15([-1,1,1,1])
0
```

16. Write a function f16(n) that will print out an upside down triangle.

```
>>> f16(3)
***
**
*
>>> f16(2)
**
*
>>> f16(1)
*
```

17. Write a function f17(list) that will print out every other element in a list in reverse order.

```
>>> f17([1,2,3,4,5,6])
6
4
2
>>> f17([1,2,3,4])
4
2
>>> f17([1])
1
```

18. Write a function f18(n) that will return n!

```
>>> f18(0)
1
>>> f18(2)
2
>>> f18(3)
6
```

19. Write a function f19(matrix) that will print the sum of each row of the matrix.

```
>>> f19([[1,0],[0,1]])
1
1
>>> f19([[1,2,3],[4,5,6]])
6
15
>>> f19([[1],[2],[3],[4]])
1
2
3
4
```

20. Write a function f20(matrix) that will print the diagonals of the matrix. Assume the matrix is a square.

```
>>> f20([[1,0],[0,1]])
1
1
>>> f20([[1,2,3],[4,5,6],[7,8,9]])
1
5
9
>>> f20([[1]])
1
```

21. Write a function f21(list) that will print the factorial of each element of a given list.

```
>>> f21([])
>>> f21([1,2,3])
1
2
6
>>> f21([1,2,3,4])
1
2
6
24
```

22. Write a function f22(list) that will print a countdown starting from each element to zero for a given list.

```
>>> f22([])
>>> f22([1,3,5])
1 0
3 2 1 0
5 4 3 2 1 0
>>> f22([5,3,6,2])
5 4 3 2 1 0
3 2 1 0
6 5 4 3 2 1 0
2 1 0
```

23. Write a function f23(list1, list2) that will return a new list where each index in the new list corresponds to list1[index] + list2[index]. Assume list1 and list2 are the same length.

```
>>> f23([], [])
[]
>>> f23([1,2,3], [1,2,3])
[2, 4, 6]
>>> f23([0,0,0], [1,2,3])
[1, 2, 3]
```

24. Write a function f24(n) that will print all the numbers from 1 to n inclusive that is a multiple of 2 or 3.

```
>>> f24(10)
2
3
4
6
8
9
10
>>> f24(1)
>>> f24(3)
2
3
```

25. Write a function f25(list) that will return the largest value in the list (of all the nested lists inside list). Note that |list| is a nested list. Assume list starts with a nonempty list.

```
>>> f25([[1,2,3],[4,5,6],[7,8,9]])
9
>>> f25([[3,2,1],[0,-1,-2]])
3
>>> f25([[1,2,3,4],[],[34],[],[],[56],[67]])
67
```

26. Write a function f26(list) that will return the second largest value in the list. Assume that the elements of list are all unique and it contains at least 2 elements.

```
>>> f26([1,4,3,2,5])
4
>>> f26([3,2])
2
>>> f26([3,4])
3
```

27. Write a function f27(n) that will return the leftmost digit in |n|. Assume |n| is positive.

```
>>> f27(1234)
1
>>> f27(4321)
4
>>> f27(3)
3
```

28. Write a function f28(list) that will print the largest value of each of the nested lists in the given list. Note that |list| is a nested list. Assume each nested list in the given list is not empty.

```
>>> f28([[1,2,3],[4,5,6],[7,8,9]])
3
6
9
>>> f28([[3,2,1],[0,-1,-2]])
3
0
>>> f28([[1,2,3,4],[1],[34],[2],[3],[56],[67]])
4
1
34
2
3
56
67
```