

National University of Singapore  
School of Computing  
CS1010S: Programming Methodology

**Extra Practice 6**

For Questions 1-6, what are the output(s) of the code?

### Question 1

```
a = [1, 2, 0, 4, 5, 6]

for i in range(len(a)):
    if a[i] % 2 == 0:
        a.pop(i)
    print(a)
```

### Question 2

```
a = []

temp = [0]*4
for i in range(3):
    a.append(temp)

a[1][1] = 99
print(a)
```

### Question 3

```
a = [1, 2, 3]
b = (1, 2, 3, a, 4, 5)
print(b)
a.clear()
print(b)
a = [1]
print(b)
```

### Question 4

```
# CS1010S AY18/19 Sem 2 Finals
a = [1, 2]
a += [a]
print(a)
b = a.copy() # shallow copy vs deep copy?
a[2] = 0
print(a)
print(b)
```

## Question 5

```
a = [1, 2, 3]
b = [4, a]
c = [a, b, 15, b[0], b[1]]
```

```
c[1][0] = 99
print(b)
print(c)
```

```
c[3] = 100
print(b)
print(c)
```

```
a[1] = 200
print(a)
print(b)
print(c)
```

```
b[1][2] = 300
print(a)
print(b)
print(c)
```

```
d = [4, c.copy()]
c[2] = 500
c[0][2] = 500
print(d)
```

## Question 6

```
# CS1010X AY18/19 Special Term I Finals
```

```
a = [4, 3, 2, 1]
b = [1, 4, 3, 2]
c = [2, 1, 4, 3]
d = [3, 2, 1, 4]
a[0], a[1], a[2] = a, c, b
b[0], b[1], b[2] = a, b, c
c[0], c[1], c[2] = c, b, d
d[0], d[1], d[2] = c, d, a
print(a[0][1][0][2][1][2][2][2][3])
```

## Question 7: Minesweeper!

In Minesweeper, we have some mines placed in a 2-D field (starting from row 0, column 0) and store the positions of these mines within a list. For instance, the grid below.

```
grid = [ ["0", "0", "X", "0"], # row 0
         ["X", "0", "X", "X"],
         ["0", "X", "0", "0"] # row 2
```

```
bombs = [[0, 2], [1, 0], [1, 2], [1, 3], [2, 1]]
```

For the position at row 1 column 2, there are a total of 4 bombs in its vicinity. "Vicinity" in minesweeper means the bombs in a 3-by-3 grid with respect to its position in the middle.

- (a) Define a function **grid\_to\_bombs** that takes in a grid as shown and returns the bombs list, containing the positions of all the bombs found in the grid (marked as "X").

**Sample Test:**

```
>>> grid_to_bombs(grid) == bombs
True
```

- (b) Now, define a function **count\_bombs** that takes in a row and a column, as well as a grid, and returns the number of bombs in the vicinity of that row and column.

**Sample Test:**

```
>>> count_bombs(1, 2, grid)
4
```