[aps]

{abertay /*PROGRAMMING*/ society;}

HackerRank Winners & Solutions

Abertay Programming Society
13/11/2018

Announcements

- 24-hour Gameathon
  - 6pm Friday - 6pm Saturday
  - £10/£11.37 tickets
  - https://www.abertaygameathon2018.co.uk/

- Discord
- Feedback
- Talks/meetings/workshops/projects/etc to present

Winners

- 3rd place: (20 points)
- 2nd place: (30 points)
- 1st place: (67.89 points)

Challenges

- Easy
  - Handshake
  - Grading Students
  - Diagonal Difference
  - Arrays - DS
  - Utopian Tree
- Medium
  - Eugene and Big Number
  - Flipping the Matrix

Handshake

n	sum	h
1	0	0
2	1	1
3	2 + 1	3
4		6
5		10
…		…
n	(n-1) + (n-2) + (n-3) + … + 2 + 1	n

$$n = \frac{1}{2} n (n-1)$$

Handshake #1 - C++

```
int handshake(int n){
    double handshakes = (n*(n-1))/2;
    return int(handshakes);
}
```

Handshake #2 - C++

```
int handshake(int n) {
    int total = 0;
    int limit = n;
    for (int i = 0; i < limit; i++) {
        total = total + (n-i);
    }
    return total;
}
```

Handshake #3 - Python3

```
def handshake(n):
    return(sum(range(n)))
```

Grading Students

Grading Students #1 - C++

```
vector<int> gradingStudents(vector<int> grades) {
    for(int i = 0; i <grades.size(); i++) {
        if(grades[i] >= 37) {
            for(int j = 1; j < 3; j++) {
                if((grades[i] + j) % 5 == 0) {
                    grades[i] = grades[i] + j;
                }
            }
        }
    }
    return grades;
}
```

Grading Students #2 - Python3

```
def gradingStudents(grades):
    out = []
    for grade in grades:
        if grade >= 38:
            if(grade % 5) > 2:
                up = 5 - grade % 5
                grade += up
        out.append(grade)
    return out
```

Grading Students #3 - Java (the best language)

```
static int[] gradingStudents(int[] grades) {
    int[] new_grades = new int[grades.length];
    for (int i=0; i<grades.length; i++){
        if (grades[i] < 38){
            new_grades[i] = grades[i];
        }
        else if ((grades[i] % 5) > 2){
            new_grades[i] = grades[i] + (5 - (grades[i] % 5));
        }
        else {
            new_grades[i] = grades[i];
        }
    }
    return new_grades;
}
```

Grading Students #4 - Python3

```
# List Comprehension
def gradingStudents(grades):
    return [((grade + (5 - grade % 5)) if ((grade >= 38) and
((grade % 5) > 2)) else grade) for grade in grades]
```

Diagonal Difference

Diagonal Difference #1 - Java

```
static int diagonalDifference(int[][] arr) {
   int matrix_size = arr.length;
   int sum_diag1 = 0;
   int sum_diag2 = 0;
   int abs_value = 0;
   for (int i=0;i<matrix_size;i++){
       int j = matrix_size - 1 - i;
       sum_diag1 = sum_diag1 + arr[i][i];
       sum_diag2 = sum_diag2 + arr[j][i];
   }
   abs_value = Math.abs(sum_diag1 - sum_diag2);
    return abs_value;
}
```

Diagonal Difference #2 - Python3

```
def diagonalDifference(arr):
    n = len(arr)
    dia_1 = sum([arr[i][i] for i in range(n)])
    dia_2 = sum([arr[i][n-1-i] for i in range(n)])
    return abs(dia_1 - dia_2)
```

Diagonal Difference #3 - Python3

```
def diagonalDifference(arr):
    return abs(sum(index[i] - index[-i-1] for i, index in enumerate(arr)))
```

Diagonal Difference #4 - C++

```
int diagonalDifference(vector<vector<int>> arr) {
    int total = 0, n = arr.size();
    for (int i =0; i < n ; i++){
        total += arr[i][n-1-i] - arr[i][i];

    }

    return abs(total);
}
```

Arrays - DS

Arrays #1 - Java

```
static int[] reverseArray(int[] a) {
    int[] new_arr = new int[a.length];
    for (int i=0; i<a.length; i++){
        new_arr[i] = a[a.length-1-i];
    }
    return new_arr;
}
```

Arrays #2 - Python3

```
def reverseArray(a):
    return [a[i] for i in range(len(a) -1, -1, -1)]
```

Arrays #3 - Python3

```
def reverseArray(a):
    a.reverse()
    return(a)
```

Arrays #4 - C++

```
#include <algorithm>

vector<int> reverseArray(vector<int> a) {
    return std::reverse(a.begin(), a.end());
}
```

Utopian Tree

Utopian Tree #1 - C++ (& Java)

```
int utopianTree(int n) {
    int height = 1;
    for (int i = 0; i < n; i++){
        if(i % 2 == 1){
            height = height + 1;
        }
        else{
            height = height * 2;
        }
    }
    return height;
}
```

Utopian Tree #2 - Python3

```
def utopianTree(n):
    height = 1
    double_cycle = True
    for i in range(n):
        if double_cycle:
            height *= 2
        else:
            height += 1
        double_cycle = not double_cycle
    return height
```

Utopian Tree #3 - Python3

```
def utopianTree(n):
    output = 1
    for i in range(0, n):
        output = output +1 if (i % 2) else output * 2
    return output
```

Any questions?