CS 1083

Assignment #8

Author: Yulong Wang

Id: 3713596

1. Source Code:

a. HexToDecimal.java

```java
import java.util.HashMap;

/**
 * Convert Hex number to decimal integer
 * @author Yulong Wang
 * @date 2021/11/05
 */
public class HexToDecimal {
    public static HashMap<Character,Integer> hexMap = new HashMap<Character,Integer>()
{{
        put('A',10);
        put('B',11);
        put('C',12);
        put('D',13);
        put('E',14);
        put('F',15);
    }};

    /**
     * @param str The hex number to be converted
     * @return int The decimal value
     * @throws IllegalArgumentException Throw if invalid input.
     */
    public static int hexToDecimal(String str) throws IllegalArgumentException{
        if(str == null){
            throw new IllegalArgumentException("No argument provided");
        }
        str = str.toUpperCase();
        if(str.length()<1){
            return 0;
        }
        char digit = str.charAt(0);
        if(digit>47 && digit<58){
            return Character.getNumericValue(digit)*(int)Math.pow(16, str.length()-1)
+ hexToDecimal(str.substring(1));
        }else if(digit>64 && digit<71){
            return hexMap.get(digit)*(int)Math.pow(16, str.length()-1) +
hexToDecimal(str.substring(1));
        }else{
            throw new IllegalArgumentException("The given string is not a hex
number");
        }
    }

    /**
     * @param args argument
     */
    public static void main(String[] args) {
        try{
            String i = args[0];
            System.out.println(hexToDecimal(i));
        }catch (ArrayIndexOutOfBoundsException e){
            System.out.println("No argument entered.");
        }catch (IllegalArgumentException e){
            System.out.println(e.getMessage());
        }
    }
}
```

b. Cavern.java

```java
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.util.LinkedList;
import java.util.Queue;

/**
 * @author Yulong Wang
 * @date 2021/10/06
 */
public class Cavern {
    /**
     * The width of the array represents the cavern
     */
    private final int width;
    /**
     * The height of the array represents the cavern
     */
    private final int height;
    /**
     * The array represents the cavern
     */
    private int[][] cavern;

    /**
     * @param height The height of the cavern
     * @param width  The width of the cavern
     * @param cavern The array that represents the cavern
     */
    public Cavern(int height, int width, int[][] cavern) {
//      assume the input is valid, this means height and width is always larger than
0;
        this.height = height;
        this.width = width;
        this.cavern = cavern;
    }

    /**
```

```
     * This method calculate and print the area of the cavern that is accessible.
     */
    public void showCavernArea(){
        int visited = 0;
        Queue<int[]> q = new LinkedList<>();
//        find the valve
        int i;
        for(i=0;i<cavern[0].length;i++){
            if(cavern[0][i] == 0){
                int[] temp = {0,i};
                cavern[0][i] = 3;
                q.offer(temp);
                visited ++;
                break;
            }
        }
        visited += findZeros(0,i);
//      print the result
        for(i=0;i<height;i++){
            for(int j=0;j<width;j++){
                System.out.print(cavern[i][j]+" ");
            }
            System.out.println();
        }
        System.out.println("The area of the cavern is: "+visited);
    }

    /**
     * @param row the row index of cavern array to find zero around
     * @param col the column index of cavern array to find zero around
     * @return int number of connected zeros found.
     */
    public int findZeros(int row, int col){
        int sum = 0;
        if(row-1>0 && cavern[row-1][col]==0){
            cavern[row-1][col] = 3;
            sum += 1 + findZeros(row-1,col);
        }
        if(row+1<height && cavern[row+1][col]==0){
            cavern[row+1][col] = 3;
            sum += 1 + findZeros(row+1,col);
        }
        if(col-1>=0 && cavern[row][col-1]==0){
            cavern[row][col-1] = 3;
            sum += 1 + findZeros(row,col-1);
        }
        if(col+1<width && cavern[row][col+1]==0){
            cavern[row][col+1] = 3;
            sum += 1 + findZeros(row,col+1);
        }
        return sum;
    };
}
```

c. TestCavernDriver.java

```java
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class TestCavernDriver {
    /**
     * @param args Test file path
     */
    public static void main(String[] args) {
        try{
            File file = new File(args[0]);
            Scanner sc = new Scanner(file);
            int row = sc.nextInt();
            int column = sc.nextInt();
            int[][] cavernArray = new int[row][column];
            sc.nextLine();
            int r = 0;
            while(sc.hasNextInt()){
                for(int i = 0; i<column;i++){
                    cavernArray[r][i] = sc.nextInt();
                }
                r ++;
            }
            Cavern cavern = new Cavern(row, column, cavernArray);
            cavern.showCavernArea();
        }catch (ArrayIndexOutOfBoundsException e){
            System.out.println("No argument provided.");
        }catch (FileNotFoundException e){
            System.out.println("Can not find the file: " +args[0]);
        }

    }
}
```

2. Sample Output:

Q1:

```
1.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal FAC
4012

2.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal fac
4012
```

```
3.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal
No argument entered.

4.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal feqwe
The given string is not a hex number

5.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal 123acdef
305843695

6.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal 0123ef
74735

7.
(base) yulongwang@promise-s-galaxy-a31 src % java HexToDecimal A01f
40991
```

Q2:

Sample Map Output

```
(base) yulongwang@afif-s-galaxy-tab-s6-lite src % java TestCavernDriver sampleMap.txt
1 1 1 1 1 3 1
1 0 0 1 3 3 1
1 1 1 3 3 3 1
1 1 3 3 1 1 1
1 0 1 3 1 3 1
1 0 1 3 3 3 1
0 0 0 1 1 1 0
1 1 1 0 0 0 1
The area of the cavern is: 13
```

map1.txt

```
7 7
1 1 1 1 1 0 1
1 0 0 0 0 0 1
1 0 0 1 1 1 1
1 0 1 0 1 0 1
1 0 0 1 0 0 1
1 0 0 0 0 0 1
1 1 1 1 1 1 1
```

Output

```
(base) yulongwang@promise-s-galaxy-a31 src % java TestCavernDriver  map1.txt
1 1 1 1 1 3 1
1 3 3 3 3 3 1
1 3 3 1 1 1 1
1 3 1 0 1 3 1
```

```
1 3 3 1 3 3 1
1 3 3 3 3 3 1
1 1 1 1 1 1 1
The area of the cavern is: 19
```

map2.txt

```
1 1
1
```

Output

```
(base) yulongwang@promise-s-galaxy-a31 src % java TestCavernDriver  map2.txt
1
The area of the cavern is: 0
```

map3.txt

```
8 7
0 1 1 1 1 1 1
0 1 1 1 1 1 0
0 0 0 0 0 0 0
0 1 1 1 1 1 0
0 1 1 1 1 1 0
0 1 1 1 1 1 0
0 1 1 1 1 1 0
0 0 0 0 0 0 0
```

Output

```
(base) yulongwang@promise-s-galaxy-a31 src % java TestCavernDriver  map3.txt
3 1 1 1 1 1 1
3 1 1 1 1 1 3
3 3 3 3 3 3 3
3 1 1 1 1 1 3
3 1 1 1 1 1 3
3 1 1 1 1 1 3
3 1 1 1 1 1 3
3 3 3 3 3 3 3
The area of the cavern is: 25
```

Eexception Handling:

```
(base) yulongwang@afif-s-galaxy-tab-s6-lite src % java TestCavernDriver notexist.txt
Can not find the file: notexist.txt
```