First Round

```java
import java.util.*;

/**
 * @author :   Tianyi Tang
 * @date :   Created in 2020-01-23 14:21
 */

class Loan {
    String name;
    Integer assetId;

    public Loan(String name, Integer assetId) {
        this.name = name;
        this.assetId = assetId;
    }

    @Override
    public String toString() {
        return "Loan{" +
                "name='" + name + '\'' +
                ", assetId=" + assetId +
                '}';
    }
}

class Asset{
    Integer id;
    List<List<String>> txns;

    public Asset (int id, List<String> strings) {
        this.id = id;
        this.txns = new ArrayList<>();
        for (int i = 0; i < strings.size() - 1; i+=2) {
            List<String> tmp = new ArrayList<>();
            tmp.add(strings.get(i));
            tmp.add(strings.get(i + 1));
            txns.add(tmp);
        }
```

```
    }

    @Override
    public String toString() {
        return "Asset{" +
                "id=" + id +
                ", txns=" + txns +
                '}';
    }
}

public class UniqueLoans {




    public List<Loan> uniqueLoans (List<Loan> loans,
List<Asset> assets) {
        HashMap<HashMap<List<String>, Integer>, Integer>
hm = new HashMap<>();
        HashMap<Integer, HashMap<List<String>, Integer>>
assetsToHm = new HashMap<>();
        for (Asset asset : assets) {
            HashMap<List<String>, Integer> tmp = new
HashMap<>();
            for (List<String> li : asset.txns) {
                tmp.put(li, tmp.getOrDefault(li, 0) +1);
            }
            hm.put(tmp, hm.getOrDefault(tmp, 0) + 1);
            assetsToHm.put(asset.id, tmp);
        }
        HashSet<Integer> hs = new HashSet<>();
        for (Integer i : assetsToHm.keySet()) {
            if (hm.get(assetsToHm.get(i)) == 1 ) {
                hs.add(i);
            }
        }
        System.out.println(hs);
        List<Loan> ans = new ArrayList<>();
        HashMap<String, Integer> loanFreq = new
HashMap<String, Integer>();
        for (Loan loan : loans) {
            loanFreq.put(loan.name,
loanFreq.getOrDefault(loan.name, 0) + 1);
```

```
        }
        //System.out.println(loanFreq);
        for (Loan loan : loans) {
            if (loanFreq.get(loan.name) == 1) {
                ans.add(loan);
            } else if (hs.contains(loan.assetId)){
                ans.add(loan);
                hs.remove(loan.assetId);
            }
        }
        return ans;
    }


    public static void main(String[] args) {
        Loan loan1 = new Loan("a" , 1);
        Loan loan2 = new Loan("a" , 1);
        Asset asset1 = new Asset(1,
Arrays.asList("hello","boa", "wealth", "blend"));
        Asset asset2 = new Asset(2,
Arrays.asList("wealth", "blend", "chase","boa"));
        List<Loan> loans = new ArrayList<>();
        loans.add(loan1);
        loans.add(loan2);
        List<Asset> assets = new ArrayList<>();
        assets.add(asset1);
        assets.add(asset2);
        System.out.println(new
UniqueLoans().uniqueLoans(loans, assets));
    }
}
```

Second Round

```
import java.util.*;


/**
 * @author :   Tianyi Tang
 * @date :   Created in 2020-01-23 16:07
 */
```

```
public class LRU {

    class Node {
        String key;
        String value;
        Node next;
        Node prev;

        public Node (String key, String value) {
            this.key = key;
            this.value = value;
            this.next = null;
            this.prev = null;
        }
    }

    public void removeNode (Node node) {
        if (node.prev == node) {
            node = null;
            first = null;
            last = null;
            return;
        }
        node.prev.next = node.next;
        node.next.prev = node.prev;
        if (node == last) {
            last = node.prev;
        } else if (node == first){
            first = node.next;
        }
        node.next = null;
        node.prev = null;
    }

    public void addNode (Node node) {
        if (first == null) {
            first = node;
            last = node;
            first.next = last;
            first.prev = last;
            return;
        }
        last.next = node;
        node.next = first;
```

```
        first.prev = node;
        first = node;
        node.prev = last;
    }

    int size;
    Node first;
    Node last;
    HashMap<String, Node> hm;

    public LRU (int size) {
        this.size = size;
        this.first = null;
        this.last = null;
        this.hm = new HashMap<>();
    }

    public void put(String key, String value) {
        Node node = new Node(key, value);
        if (hm.containsKey(key)) {
            removeNode(hm.get(key));
            hm.put(key, node);
            addNode(node);
            return;
        }
        hm.put(key, node);
        addNode(node);
        if (hm.size() > size) {
            hm.remove(last.key);
            removeNode(last);
        }
    }

    public String get(String key){
        if (!hm.containsKey(key)) {
            return null;
        }
        Node node = hm.get(key);
        System.out.println(node.key);
        System.out.println(node.prev.key);
        System.out.println(node.next.key);
        removeNode(node);
        addNode(node);
        return node.value;
```

```
    }

    public void print () {
        HashSet<String> seen = new HashSet<>();
        Node cur = first;
        while (!seen.contains(cur.next.key)) {
            System.out.println(cur.key + "  "+
cur.value);
            cur = cur.next;
            seen.add(cur.key);
        }
    }


    public static void main(String[] args) {
        LRU lru = new LRU(3);
        lru.put("abc","abc");
        //lru.print();
        lru.put("a","a");
        //System.out.println(lru.hm);
        //lru.print();
        System.out.println(lru.get("a"));
        lru.put("c", "d");
        lru.put("d","s");
        System.out.println(lru.get("abc"));
    }


}
```

Third Round

```
import java.util.*;
/**
 * @author :   Tianyi Tang
 * @date :   Created in 2020-01-23 17:42
 */
public class MineSwepper {

    char[][] userBoard;
    char[][] actualBoard;
```

```
    boolean gg = true;

    public void init(int length, int width, List<int[]>
mines) {
        userBoard = new char[length][width];
        actualBoard = new char[length][width];
        for (int i = 0; i < length; i++) {
            for (int j = 0; j < width; j++){
                userBoard[i][j] = 'X';
                actualBoard[i][j] = '0';
            }
        }
        for (int[] mine : mines) {
            int x = mine[0];
            int y = mine[1];
            actualBoard[x][y] = 'M';
            for (int i = -1; i < 2; i++) {
                for (int j = -1; j < 2;j++) {
                    int tempX = x + i;
                    int tempY = y + j;
                    if (tempX >= 0 && tempX < length &&
tempY >= 0 && tempY < width && actualBoard[tempX][tempY]
!= 'M') {
                        actualBoard[tempX][tempY] =
(char)(actualBoard[tempX][tempY] + 1);
                    }
                }
            }
        }
        printActual();
    }


    public void dfs (int x, int y) {
        if (actualBoard[x][y] == 'M') {
            userBoard[x][y] = 'M';
            //printUser();
            System.out.println("Game Over!");
            gg = false;
            return;
        }
        if (userBoard[x][y] != 'X') return;
        if (actualBoard[x][y] != '0') {
            userBoard[x][y] = actualBoard[x][y];
```

```
            return;
        }
        userBoard[x][y] = actualBoard[x][y];
        for (int i = -1; i < 2; i++) {
            for (int j = -1; j < 2; j++) {
                int tempX = x + i;
                int tempY = y + j;
                if (tempX >= 0 && tempX <
userBoard.length && tempY >= 0 && tempY <
userBoard[0].length) {
                    dfs(tempX, tempY);
                }
            }

        }
    }

    public void click(int x, int y) {
        if (!gg) {
            System.out.println("You cannot play
anymore");
            return;
        }
        dfs(y, x);
        printUser();
    }

    public void printUser () {
        for (char[] chars : userBoard) {
            System.out.println(Arrays.toString(chars));
        }
    }

    public void printActual () {
        for (char[] chars : actualBoard) {
            System.out.println(Arrays.toString(chars));
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        List<int[]> mines = new ArrayList<>();
        mines.add(new int[]{2, 3});
        mines.add(new int[]{2, 4});
```

```
        MineSwepper mineSwepper = new MineSwepper();
        mineSwepper.init(6,8, mines);
        System.out.println("---------Input your
location, with space to split x and y");
        while (scanner.hasNext()) {
            String[] strs = scanner.nextLine().split("
");
            int x = Integer.valueOf(strs[0]);
            int y = Integer.valueOf(strs[1]);
            mineSwepper.click(x, y);
            if (!mineSwepper.gg) return;
        }
    }
}
```

Four Round

```
import java.util.*;
/**
 * @author :   Tianyi Tang
 * @date :   Created in 2020-01-23 18:42
 */
public class Parse {


    //( add 2 3 () )
    public int parse (String string) {
        if (Character.isDigit(string.charAt(0))) {
            return Integer.valueOf(string);
        }
        if (string.length() == 3 ) {
            return 0;
        }
        int firstSpace = string.indexOf(' ');
        int secondSpace = string.indexOf(' ', 3);
        String sub = string.substring(firstSpace + 1,
secondSpace);
        String str = string.substring(secondSpace + 1,
string.length() - 1);
        List<String> list = readString(str);
        int ans = 0;
```

```
        switch (sub) {
            case "add" :
                for (String s : list) {
                    ans += parse(s);
                }
                break;
            case "sub" :
                ans = parse(list.get(0));
                for (int i = 1; i < list.size(); i++) {
                    ans -= parse(list.get(i));
                }
                break;
            case "mult" :
                ans = parse(list.get(0));
                for (int i = 1; i < list.size(); i++) {
                    ans *= parse(list.get(i));
                }
                break;
            case "div" :
                ans = parse(list.get(0));
                for (int i = 1; i < list.size(); i++) {
                    ans /= parse(list.get(i));
                }
            default:
                break;
        }
//        if (sub.equals("( a")) {
//            for (String s : list) {
//                ans += parse(s);
//            }
//        } else if(sub.equals("( s")) {
//            ans = parse(list.get(0));
//            for (int i = 1; i < list.size(); i++) {
//                ans -= parse(list.get(i));
//            }
//        } else if (sub.equals("( m")) {
//            ans = parse(list.get(0));
//            for (int i = 1; i < list.size(); i++) {
//                ans *= parse(list.get(i));
//            }
//        } else {
//            ans = parse(list.get(0));
//            for (int i = 1; i < list.size(); i++) {
//                ans /= parse(list.get(i));
```

```
//            }
//        }
        return ans;
    }

    public List<String> readString (String str) {
        char[] cs = str.toCharArray();
        List<String> res = new ArrayList<>();
        StringBuilder sb = new StringBuilder();
        int p = 0;
        for (char c : cs) {
            if (c == '(') p++;
            if (c == ')') p--;
            if (c == ' ' && p == 0) {
                res.add(sb.toString());
                sb = new StringBuilder();
                continue;
            }
            sb.append(c);
        }
        if (sb.length() != 0) {
            res.add(sb.toString());
        }
        return res;
    }

    public static void main(String[] args) {
        Parse parse = new Parse();
        System.out.println(parse.parse("( add ( ) 3 5 (
mult 3 5 ( div 14 7 ) ) )"));
    }
}
```