1 Syntax Mastery (Spring 2015 Final Q5)

Give the output of `main`. You may not need all lines.

```java
public class Sklarp implements Iterable<Character>, Iterator<Character> {
    public char[] contents;
    public char magicCharacter;
    public int k;
    public Sklarp(char[] s, Character c) {
        contents = s;
        magicCharacter = c;
        k = 0;
    }
    public Iterator<Character> iterator() {
        return this;
    }
    public boolean hasNext() {
        return k < contents.length;
    }
    public Character next() {
        if (k % 3 == 0) {
            contents[k] = magicCharacter;
        }
        char returnChar = contents[k];
        k += 1;
        return returnChar;
    }
    public void remove() {
        throw new UnsupportedOperationException();
    }
    public static void main(String[] args) {
        Sklarp g = new Sklarp("Zeoidei".toCharArray(), 'r');
        for (Character c : g) {
            System.out.print(c);
        }
        System.out.println();
        for (Character c : g) {
            System.out.print(c);
        }
    }
}
```
For each of the pieces of code below, give the runtime in $\Theta(\cdot)$ notation as a function of $N$. Your answer should be simple, with no unnecessary leading constants or unnecessary summations.

```java
public static void p1(int N) {
    for (int i = 0; i < N; i += 1) {
        for (int j = 1; j < N; j = j + 2) {
            System.out.println("hi !");
        }
    }
}

P1 answer: $\Theta(\cdot)$

```java
public static void p2(int N) {
    for (int i = 0; i < N; i += 1) {
        for (int j = 1; j < N; j = j * 2) {
            System.out.println("hi !");
        }
    }
}

P2 answer: $\Theta(\cdot)$

```java
public static void p3(int N) {
    if (N <= 1) return;
    p3(N / 2);
    p3(N / 2);
}

P3 answer: $\Theta(\cdot)$

```java
public static void p4(int N) {
    int m = (int)((15 + Math.round(3.2 / 2)) *
                  (Math.floor(10 / 5.5) / 2.5) * Math.pow(2, 5));
    for (int i = 0; i < m; i++) {
        System.out.println("hi");
    }
}

P4 answer: $\Theta(\cdot)$

```java
public static void p5(int N) {
    for (int i = 1; i <= N * N; i *= 2) {
        for (int j = 0; j < i; j++) {
            System.out.println("moo");
        }
    }
}

P5 answer: $\Theta(\cdot)$
```
Suppose we have a HashMap, but want to be able to undo operations made on it. Implement HistoryMap below to have this functionality. The only operations that we care about that modify the structure are put and remove.

Calling undo should revert the state of the HistoryMap to before the last put or remove, whichever was most recent. See the main method for example behavior. Assume remove is used correctly; any key removed is assumed to already exist in the HistoryMap. You may not need all lines.

[Hint: Use Java’s built-in Stack<E> class, which has methods push and pop.]

```java
public class HistoryMap<K, V> extends HashMap<K, V> {
    class ___________________________________ { /* Helper class */ /* Place fields/variables here */
        /* Place the constructor here */
        (______________________________________________) {
            ______________________________________________
            ______________________________________________
            ______________________________________________
        }
    }

    @Override
    /** Remember that in a HashMap, a null value is valid */
    public V put(K key, V value) {
        ______________________________________________
    }
    @Override
    public V remove(Object key) {
        ______________________________________________
    }
    // Continues on next page
```
@Override
public boolean containsKey(K key) {
    return super.containsKey(key);
}

public void undo() {
    if (_____________________________________________________) {
        return;
    }
    if (_____________________________________________________) {
        ____________________________________________
    } else {
        ____________________________________________
    }
}

public static void main(String[] args) {
    HistoryMap<String, Integer> h = new HistoryMap<>();
    h.put("party", 1);
    h.put("parrot", 2);
    h.put("conga", 4);
    h.put("parrot", 3);
    h.undo();
    h.undo();
    System.out.println(h); // Output: {parrot=2, party=1}
    h.remove("party");
    h.undo();
    System.out.println(h); // Output: {parrot=2, party=1}
}