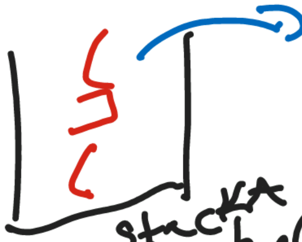
Whiteboard 13

WHITE BOARD13

Take a string as only argument into a boolean method. The sting takes braces { }, brackets [ ] and parenths + other charaters and returns true if each bracket type is balanced in a given String, i.e. a (has a) somewhere in the string. Be sure to include 3 tests per method.

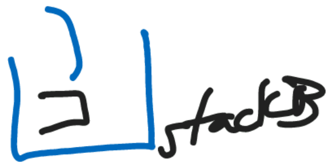
Visual

if take in a stack



pop & store in checker stack

if stack empty false



if popped value doesn't have match keep popping

if match never found return false

else return true

Big O

$O(n^2)$ time
$O(n)$ space > before refactoring

PseudoCode

```
1) if stack empty
      return false                1.5 inst. empty stacks ↓

2) sort existing stack into
      up to 4 stacks using loop
   2.A while(original stack
         size > 0)
      2.A.1 if ("{" == "}" || "[" == "]")
            put in curly stack
      2.A.2 else if "(" or ")"
            put in parenth stack
```

```
2.A.3 else if ("[" or "]")
        put in bracketStack

2.A4 esle put in
        otherStack

helper method
    3) compare size of each stack
3.A.1 if curly stack size
        is divisible by 2 = even
        ~~return true goto next~~
      & if bracket stack
        is divisible by 2 = even
      & if parenths stack
        is divisible by 2 = even

        and @ least one
        stack size isn't empty
      return true
3.B.1 Else return false
```

So, restated,

1. take instring
2. turn string into stack
3. instatiate stacks
   - curly stack for only {}
   - bracket stack for only [ ]
   - parenth stack for only ( )
   - other stack for all/any other characters
4. Have a sorting method to go through original stack (if not empty) and fill other stacks

⑤ Have a "pair checking" method that sees that @ least one of 4 new stacks size is > 0 then checks if any of the stacks' size is odd if odd return false else return true

Tests to Consider

1. Empty input stack
2. Empty stacks for specialty stacks (ie charactersonly)
3. Only one specialty stack filled
   - A) Even size true
   - B) Odd size false
4. Multi specialty Stacks full
   - A) Some Odd size
   - B) All Odd size
   - C) None " "

Code

```
public class BracketValidation {
    private Stack original;
      "       "     curly;
      "       "     parens;
      "       "     bracket;
      "       "     other;

    public static boolean
        multiBracketValidation
        (String input){
        original = (stack) input;
          bracketSort (orginal);


    }
```

```
public boolean bracketSort
    (Stack original){
    int size = original.size();
    if(   size   == Ø){
        return false;
    }
    while(size > 0){
      String temp = original.pop;
      if(temp == "{" || temp ==
          "}"){
          curly.push(temp);
      } else if(temp == "[" ||
          temp == "]"){
          bracket.push(temp);
```

```
} else if(temp=="(" ||
  temp==")" {
    parenths.push(temp);
 } else(temp!="(" &&
        temp!=")" &&
        temp!="]" &&
        temp!="[" &&
        temp!="{" &&
        temp!="}") {
    other.push(temp
  }
}
if(curly.maached()==
   true && bracket.matched()
  == true && parenths.matched()
   = true){
```

```
  reten true;
 } else {
   return false;
  }
}
public boolean matched(){
boolean curlysize = (curly.size
      %2 = =0);
 boolean bracketsize = (bracket.size
      %2==0);
  boolean parenthsize =
     (parenthe. size %2
        = =0);
 while(matchedfalse){
  }
```

```
// end program
```