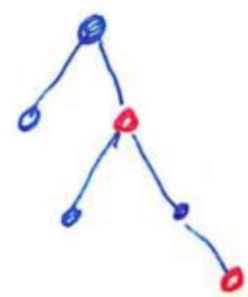
Lab 8 solutions

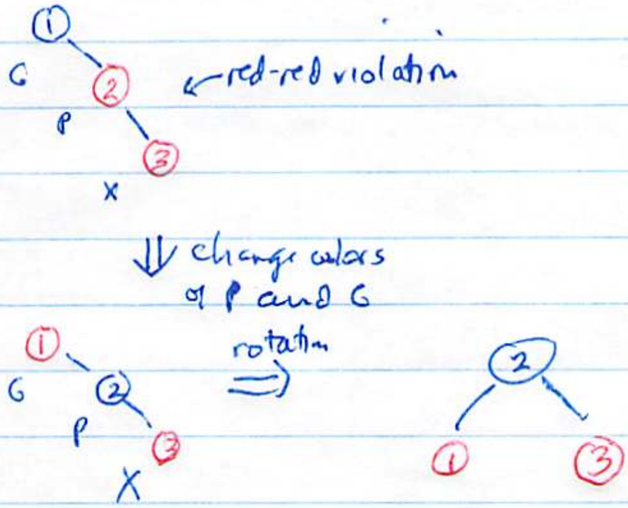
Problem 1

A red-black tree that is not AVL

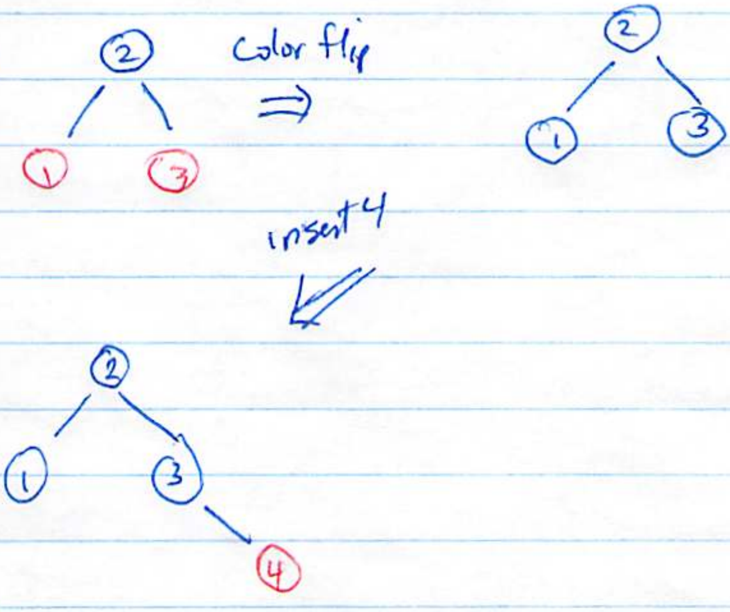


3 a insertion sequence 1, 2, 3, 4, 5, 6, 7, 8

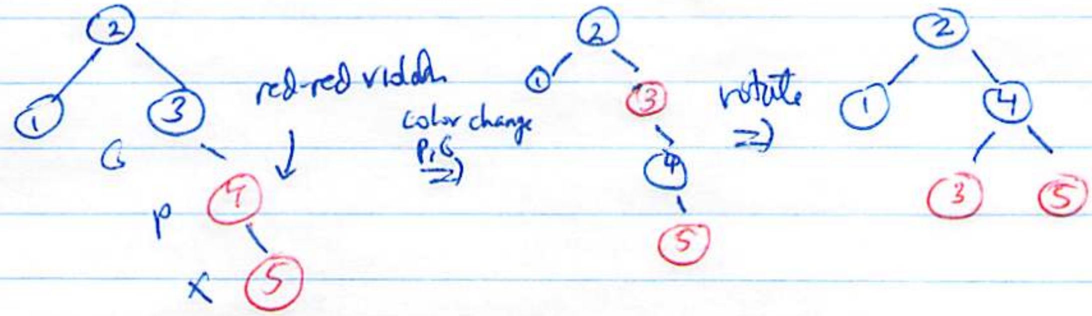
begin insert 3



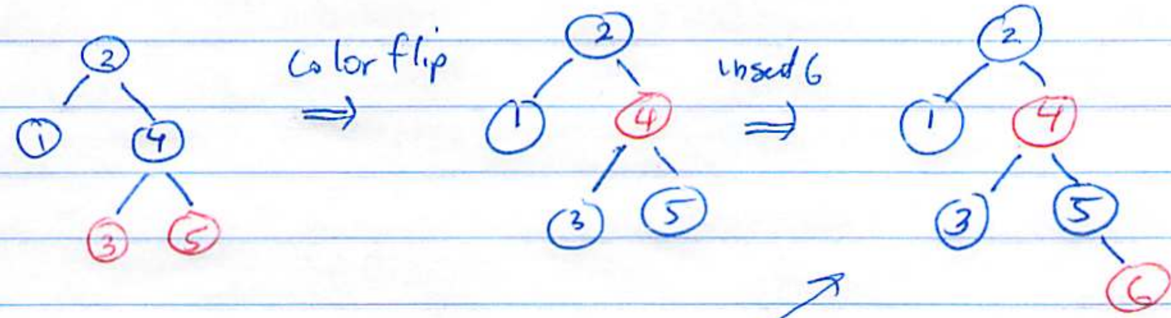
begin insert 4



begin insert 5

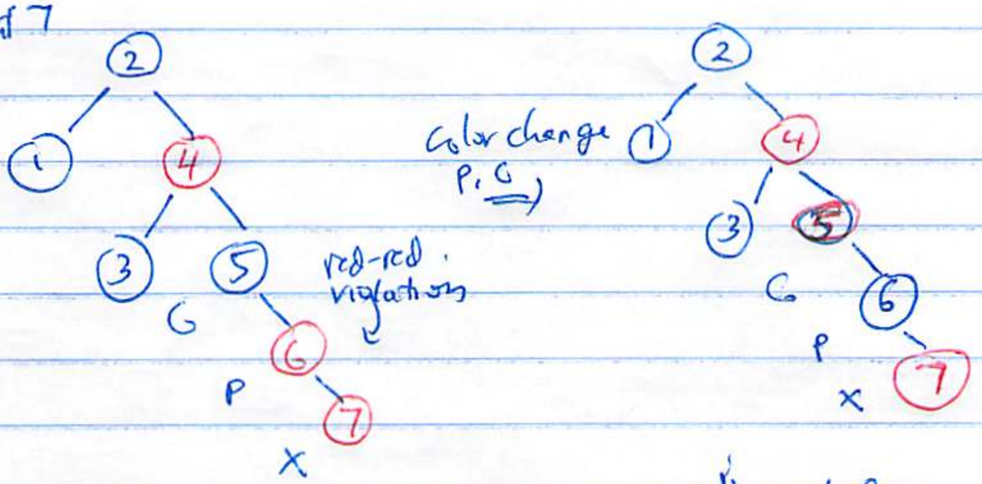


begin insert 6 ⇒ color flip ⇒ insert 6 ⇒

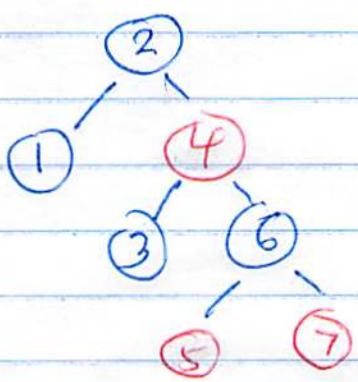


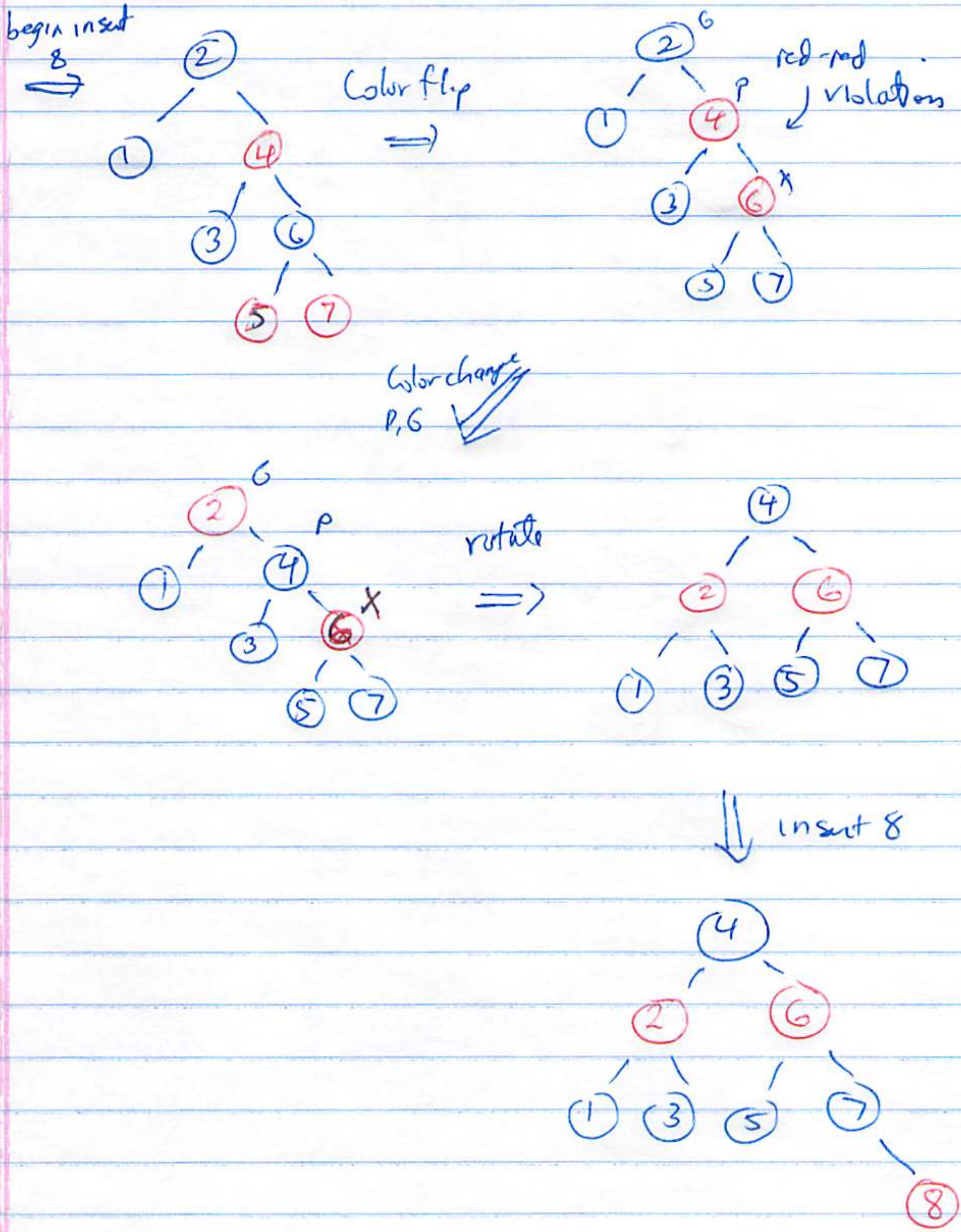
[Note: The RB tree is not AVL.
Show there is a derivable RB tree
that is not AVL]

begin insert 7 ⇒



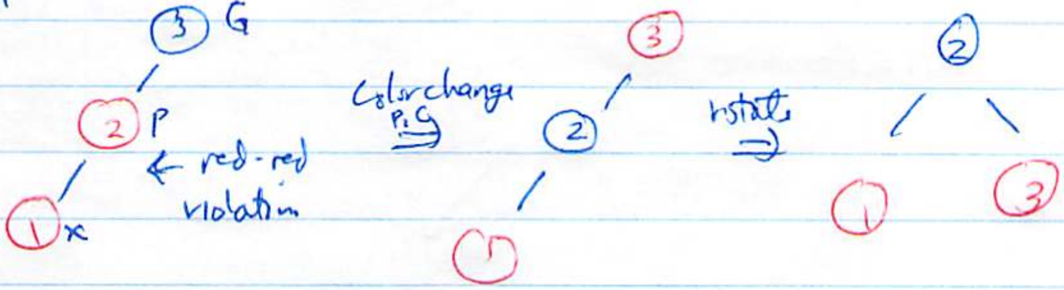
⇓ rotate



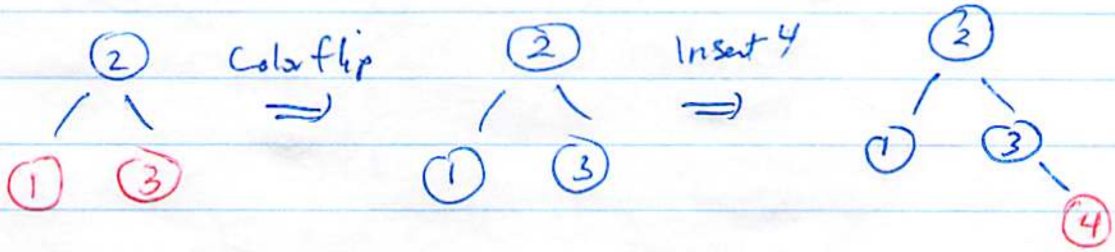

begin insert 8
Color flip
red-red violation
Color change P,G
rotate
insert 8

3b insertion sequence 3, 2, 1, 4, 5, 6

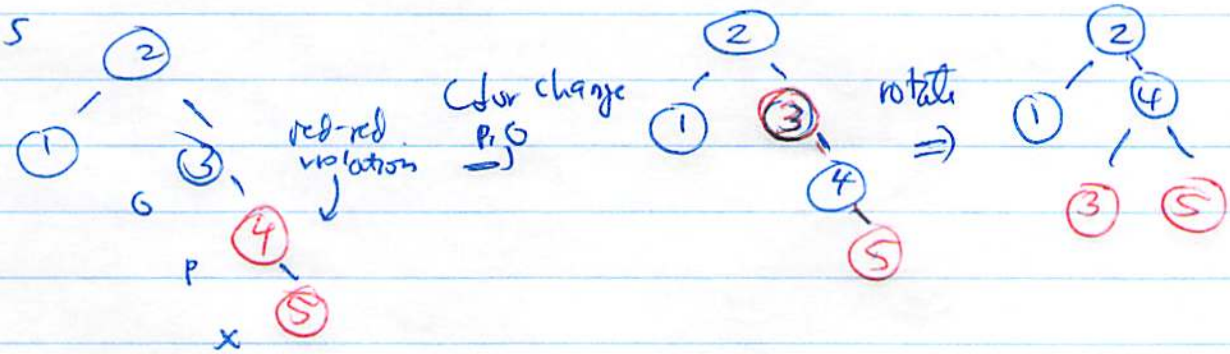
begin insert 1 ⟹



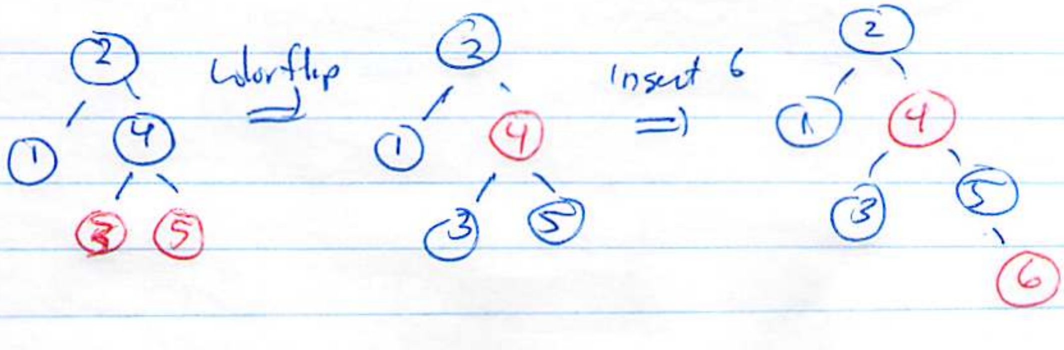
begin insert 4 ⟹



begin insert 5 ⟹



begin insert 6 ⟹



```java
package primes;

public class IsPrime {
    //Precondition: n is a positive integer
    public static boolean isPrime(int n) {
        if(n == 2) return true;
        if(n== 1 || n % 2 == 0) return false;
        //check the odd numbers <= sqrt(n)
        for(int i = 3; i * i <= n ; i = i+2) {
            if(n % i == 0) return false;
        }
        return true;
    }

    public static void main(String[] args) {
        for(int i = 1; i <= 500; ++i) {
            if(isPrime(i)) System.out.println(i + " is prime");
            else System.out.println(i + " is composite");
        }
    }
}
```

Lab 8, Prob 4, Solution

Problem 4.

(A) Express the asymptotic running time of your algorithm `IsPrime(n)` in terms of the input size rather than input value.

Solution. The running time in terms of input value is given by $T(n) = n^{\frac{1}{2}}$. Since the number b of bits in a positive integer n is $\Theta(\log n)$, it follows that n is $\Theta(2^b)$ and so running time in terms of b is given by

$$T(b) = \Theta(\left(2^b\right)^{\frac{1}{2}}) = \Theta(2^{\frac{b}{2}}).$$

(B) Suppose $T(b)$ is the running time of your algorithm in terms of input size. Show that b^2 is $o(T(b))$. (It can be shown that b^k is $o(T(b))$ for any positive integer k. Consequently, this algorithm is said to run in *superpolynomial* time.)

Solution. Using L'Hopital twice we have:

$$\begin{aligned}
\lim_{n\to\infty} \frac{b^2}{2^{\frac{b}{2}}} &= \lim_{n\to\infty} \frac{2b}{\frac{1}{2}\cdot 2^{\frac{b}{2}} \cdot \ln 2} \\
&= \lim_{n\to\infty} \frac{2}{\frac{1}{4}\cdot 2^{\frac{b}{2}} \cdot \ln^2 2} \\
&= 0.
\end{aligned}$$

(C) Do you think there could be an algorithm for determining whether an integer is prime that runs in polynomial time (relative to the input size)?

Solution. Answer is Yes – this was discussed in Lesson 15 slides.