Answer Pages

Question 21 (`pushAll`) answer:

```
void Tree::pushAll(Treenode * n){
    if(n == NULL){
      return;}
      st.push(n);
      pushAll(n->left);

}
```

Question 22 (`KStep`) answer:

```
KStep::KStep(){
    pushAll(root);
}
```

Question 23 (`hasMore`) answer:

```
bool KStep::hasMore(){
    if (st.isEmpty())
        return false;
    else
        return true;

}
```

Question 24 (`step1`) answer:

```
int KStep::step1(){
    Treenode* temp = st.pop();
    if (temp -> right != NULL){
        pushAll(temp -> right);
    }

    return temp -> data;

}
```

Question 25 ~~(step1 running time)~~ step(k) answer:

```
int KStep :: step (int k) {
    int temp = st.peep()->data;
    int t = 0
    for (int i=0 ; i<k; i++) {
        if (! hasMore ())
        {return temp;}
        t = step1();
    }
    return temp;
}
```

Question 26 answer:

Lower Bound	O(1)
Average	O(log n)
Upper Bound Case	O(log n)

Assuming there is constructor for QuadtreeNode

Question 27 (`buildPerfectTree`) answer:

```
QuadtreeNode* Quadtree::buildPerfectTree(int k, RGBAPixel p){
    QuadtreeNode* seed = new QuadtreeNode();
    if(root == NULL){ root = seed;}
    if (k == 1){
        seed -> element = p;
    }
    else{
        seed -> nwChild = buildPerfectTree(k-1, p);
        seed -> swChild = buildPerfectTree(k-1, p);
        seed -> neChild = buildPerfectTree(k-1, p);
        seed -> seChild = buildPerfectTree(k-1, p);
        seed -> element = p;
    }
    return seed;

}
```

Question 28 (`perfectify`) answer: TA said I can use helper functions!!

```
Void Quadtree::perfectify(int levels){
    RGBApixel temp = root -> element;

    perfectify (root, levels, temp);

}

Void Quadtree::perfectify (QuadtreeNode* node, int lvls, RGBAPixel p){
    if (node == NULL){
        node == buildPerfectTree(lvls, p);
    }else{
        perfectify (node -> nwChild, lvls - 1, p);
        perfectify (node -> neChild, lvls - 1, p);
        perfectify (node -> swChild, lvls - 1, p);
        perfectify (node -> seChild, lvls - 1, p); }
}
```

if(lvls == 0)
return;

copy paste here
(TA approved)

1 1
2 5
3 21

Question 29 (`perfectify` running time) answer:

$n^2 = 4^{(h-1)} + 1 = 2^{2(h-1)}$ $O(h \log h)$???

$(\log_4 n = h-1)$

$h = \log n$

Question 30 answer:

You may answer this question by filling in these blanks, or use the blank space for your own proof/disproof.

Preliminaries Let $H(n)$ denote the maximum height of an n-node SAVL tree, and let $N(h)$ denote the minimum number of nodes in an SAVL tree of height h. To prove (or disprove!) that $H(n) = \mathcal{O}(\log n)$, we attempt to argue that

$h \leq 3\log_2 n$

$$H(n) \leq 3\log_2 n, \text{ for all } n$$

$\frac{h}{3} \leq \log_2 n$

$2^{\frac{h}{3}} = n$

$1 + 4 + 7$

Rather than prove this directly, we'll show equivalently that

a) $N(h) \geq$ $2^{h/3}$, (1pt)

Proof For an arbitrary value of h, the following recurrence holds for all SAVL Trees:

b) $N(h) =$ 1 $+$ $N(h-1)$ $+$ $N(h-2)$, (2pt)

c) and $N(0) =$ 1, $N(1) =$ 2, $N(2) =$ 4, (2pt)

$N(3) = 7$ $N(4) = 12$ $N(5) = 20$

We can simplify this expression to the following inequality, which is a function of $N(h-3)$:

d) $N(h) \geq$ $N(h-1)$ $\times$ $2^{(1/3)}$, (1pt)

$N(h-3)$
$= 1 + N(h-4) + N(h-5)$

By an inductive hypothesis, which states:

e) for $k \geq h$ $N(k) = 2^{\left(\frac{k-1}{3}\right)} \cdot 2^{1/3} = 2^{k/3}$, (1pt)

2^{h-1} $+1$

we now have

f) $N(h) \geq$ $2^{h/3}$ = part (a) answer, (1pt)

which is what we wanted to show.

Given that $2^0 = 1, 2^{1/3} \approx 1.25$, and $2^{2/3} \approx 1.58$, what is your conclusion?

Is an SAVL tree $\mathcal{O}(\log n)$ or not? (Circle one): (2pt)

YES NO

$N(h-3) = N(h-2) \times N(h-1)$

$N(h) = 2^{(h-1)/3} \times ?$

$2^{h/3} = 2^{(h-1)/3} \cdot 2^{\left(\frac{1}{3}\right)}$

$2^{h/3} = 2^{(h-1)/3} \cdot x$

$h/3 = (h-1/3) \cdot \log x$

$h = h-1 \cdot \log x$

$h = h\log x - \log x$

Overflow Page

Use this space if you need more room for your answers.