HW3

1. $Pr[kth\ node\ has\ one\ child] = \frac{number\ of\ the\ cases\ where\ kth\ node\ has\ one\ child}{total\ number\ of\ cases}$

   There are 6 ways to insert x[k-1], x[k], and x[k+1] is they exists

   For kth value to have one child, we need to make sure the following inserted order:

   …x[k-1]...x[k]...x[k+1]... OR

   …x[k+1]...x[k]...x[k-1]...

   The other 4 inserted order will not work:

   …x[k-1]...x[k+1]...x[k]... OR

   …x[k]…x[k-1]… x[k+1]… OR

   …x[k]… x[k+1]… x[k-1]… OR

   …x[k+1]… x[k-1]… x[k]…

   $Pr[kth\ node\ has\ one\ child] = \frac{2}{6} = \frac{1}{3}$

   If x[k] is the smallest and largest value, then it must be the first inserted value to have a child. Otherwise it will be the leaf node.

   $Pr[kth\ node\ has\ one\ child] = \frac{1}{n}$

2. For B-Tree with order m, all nodes have at most *m* data items and at most *(m+1)* child pointers. Then we have $8 * m + 4 * (m + 1) \leq 512$, so the order $m \leq 42$

3. In a 51-101 B-tree, each node has [51, 101] children and has [50, 100] keys. Since we are inserting value in sorted order, only the rightmost child of the node will be affected. All other nodes except the root will stay the same after creation. To get height 3, we have the following…

   Level 2 is the root and it has 100 keys and 101 children

   Level 1, only the rightmost child is full and has 100 keys and 101 children. The remaining 100 nodes each have 50 keys and 51 children, which is 5000 keys and 5100 children. For level 1, we have 5100 keys and 5201 children.

   Level 0 (nodes with null children), only the right most child is full and has 100 keys. The remaining 5200 children only have 50 keys. For level 0, we have 100 + 5200 * 50 = 260100 keys

   In sum, we have (100 + 5100 + 260100) = 265300 keys before we get height 3 by inserting the next value. The inserted value causing height 3 is 265301.

4. Since 1,2,3 have just been searched, they are located near the root. There are 3 possible shapes: