

Question1:

Set the original stack size 200, to run 500 iterations, the stack will double size twice.

```
*/  
public static void main(String[] args) {  
    // TODO code application logic here  
    Stack stack = new Stack(200);  
    for (int i = 1; i <= 500; i++) {  
        stack.arry = stack.push(i);  
    }  
    stack.display();  
}
```

run:

```
Stack is full, the array will double sized  
Stack is full, the array will double sized  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11
```

```
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500
```

Question2:

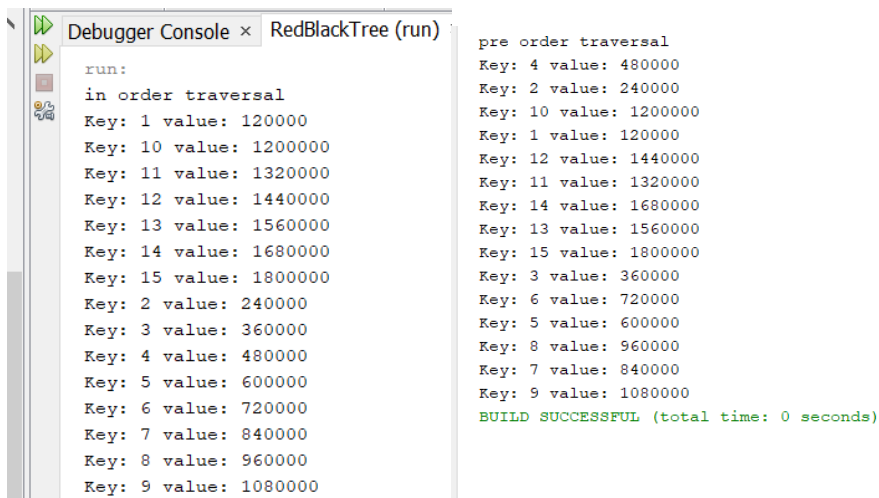
Create Map to store info

Insert key 1 to 15, and their corresponding values into tree

Print with inorder, preorder

```
public static void main(String[] args) {
    // TODO code application logic here
    RedBlackTree tree = new RedBlackTree();

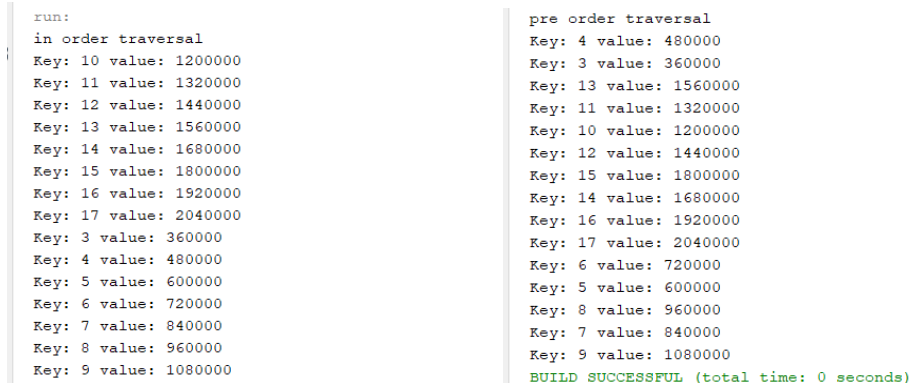
    //create the map for keyset and values
    Map<String, BigInteger> test = new HashMap<String, BigInteger>();
    for (int i = 0; i < 20; i++) {
        String key = String.valueOf(i);
        String value = String.valueOf(i * 120000);
        test.put(key, new BigInteger(value));
    }
    //insert key into tree based on values
    for (int i = 1; i < 16; i++) {
        String key = String.valueOf(i);
        tree.insert(key, test.get(key));
    }
    //print inOrder
    System.out.println("in order traversal");
    tree.inOrderTraversal(tree.root);
    //print preOrder
    System.out.println("pre order traversal");
    tree.preOrderTraversal(tree.root);
}
```



```
run:
in order traversal
Key: 1 value: 120000
Key: 10 value: 1200000
Key: 11 value: 1320000
Key: 12 value: 1440000
Key: 13 value: 1560000
Key: 14 value: 1680000
Key: 15 value: 1800000
Key: 2 value: 240000
Key: 3 value: 360000
Key: 4 value: 480000
Key: 5 value: 600000
Key: 6 value: 720000
Key: 7 value: 840000
Key: 8 value: 960000
Key: 9 value: 1080000

pre order traversal
Key: 4 value: 480000
Key: 2 value: 240000
Key: 10 value: 1200000
Key: 1 value: 120000
Key: 12 value: 1440000
Key: 11 value: 1320000
Key: 14 value: 1680000
Key: 13 value: 1560000
Key: 15 value: 1800000
Key: 3 value: 360000
Key: 6 value: 720000
Key: 5 value: 600000
Key: 8 value: 960000
Key: 7 value: 840000
Key: 9 value: 1080000
BUILD SUCCESSFUL (total time: 0 seconds)
```

Other cases, key from 3-17



```
run:
in order traversal
Key: 10 value: 1200000
Key: 11 value: 1320000
Key: 12 value: 1440000
Key: 13 value: 1560000
Key: 14 value: 1680000
Key: 15 value: 1800000
Key: 16 value: 1920000
Key: 17 value: 2040000
Key: 3 value: 360000
Key: 4 value: 480000
Key: 5 value: 600000
Key: 6 value: 720000
Key: 7 value: 840000
Key: 8 value: 960000
Key: 9 value: 1080000

pre order traversal
Key: 4 value: 480000
Key: 3 value: 360000
Key: 13 value: 1560000
Key: 11 value: 1320000
Key: 10 value: 1200000
Key: 12 value: 1440000
Key: 15 value: 1800000
Key: 14 value: 1680000
Key: 16 value: 1920000
Key: 17 value: 2040000
Key: 6 value: 720000
Key: 5 value: 600000
Key: 8 value: 960000
Key: 7 value: 840000
Key: 9 value: 1080000
BUILD SUCCESSFUL (total time: 0 seconds)
```

Question 3:

[illegible]

```
Debugger Console x RedBlackTree (run) x
run:
Please enter the operations below, Enter nothing and press enter to quit
x 4 =
4
y 5 =
5
x y +
9
x x 20 + =
24
lowerVal 1 =
1
upperVal 10 =
10
interval upperVal lowerVal - 1 + =
10
a 4 = 2 +
6
a
4

BUILD SUCCESSFUL (total time: 1 minute 44 seconds)
```

```
run:
Please enter the operations below, Enter nothing and press enter to quit
a 1 =
1
a b +
Exception in thread "main" java.lang.Exception: error: no variable b
BUILD SUCCESSFUL (total time: 8 seconds)
```

```
run:
Please enter the operations below, Enter nothing and press enter to quit
2 3 #
Exception in thread "main" java.lang.Exception: error: stack underflow exception
BUILD SUCCESSFUL (total time: 4 seconds)
```

```
run:
Please enter the operations below, Enter nothing and press enter to quit
3 4 =
3 is not a variable
BUILD SUCCESSFUL (total time: 2 seconds)
```