(1). Derive a recurrence relation and give initial conditions for the number of bit strings of length n without two consecutive 0s.

Ans:

Suppose you had a bit string of size $n = 1$. There are two possible strings:

`0` `1`

For a bit string of size $n = 2$, we have the following strings:

01 11 10

In general, we are concerned with the number of ways which we can choose the last character in the string, and the choice of character.

- If the last character was a 0, we may *only* choose the next character to be a 1.
- It follows that if the last character was a 0, then the character before the last character *must have been a 1.*
- Otherwise, we may choose either 1 or 0 as our next character.

Then suppose that $T(n)$ represents the number of ways to form a bit string of length n. From our observations above, we know that $T(n)$ is composed of the number of ways to form a string of length $n-1$ ending in 1, and a string of length $n-1$ ending in 0.

However, we also know that a string ending in 0 is simply a string ending in 1 with a 0 appended to its end.

In essence, we are looking for the number of ways to form a string of length $n-1$ ending in 1, and a string of length $n-2$ ending in 1.

$$T(n) = T(n-1) + T(n-2)$$

$$T(1) = 2\,, T(2) = 3$$

(2). The number of bacteria in a colony doubles every hour. If the colony begins with 5 bacteria, denoted as $B(0) = 5$, how many will be present in n hours, $B(n)$?

Ans:

$$\begin{aligned} B(n) &= 2(B(n-1)), B(0) = 5 \\ &= 5 \cdot 2^n \end{aligned}$$

(3). A rock band would like to tour n cities. However, time will allow for visits to only k cities. Because time is short, the band members are not concerned about the order in which they visit the same k cities. Let $g(n, k)$ be the number of groups of k cities chosen from n. Derive $g(n, k)$ with three base cases of $k = 0, k = n$, and $k > n$.

Ans:

Consider a single city. The band may visit it, in this case, they will choose $k - 1$ remaining cities from $n - 1$, or they may not visit it, and they will choose k cities from $n - 1$.

$$g(n, k) = g(n - 1, k - 1) + g(n - 1, k)$$

Where if $k = 0$, there is one way to choose 0 cities, if $k = n$, there is only one way to choose all the cities, and if $k > n$, there are zero ways to choose more than n cities.

(4). Write a recursive function to determine whether two trees are the same in terms of the data they store and the order in which they store it.

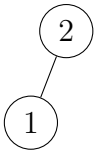
```
struct Tree {
    int key;
    Tree *left;
    Tree *right;
};
bool equal_trees(Tree *tree1, Tree *tree2){
    if(tree1 ^ tree2) return !(tree1 || tree2);
    if(tree1->key != tree2->key) return false;
    bool left = true;
    if(tree1->left && tree2->left){
        left = equal_tree(tree1->left, tree2->left);
    }
```

```
    else{
        left = !(tree1->left || tree2->left);
    }
    bool right = true;
    if(tree1->right && tree2->right){
        right = equal_tree(tree1->right, tree2->right);
    }
    else{
        right = (tree1->right || tree2->right);
    }
    return (left && right);
}
```

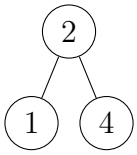
(5). Show, number-by-number, the resulting AVL trees of inserting 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty AVL tree. Denote either single-rotation or double-rotation was used to rebalance the tree.

2:

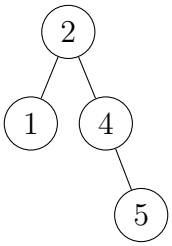
1:



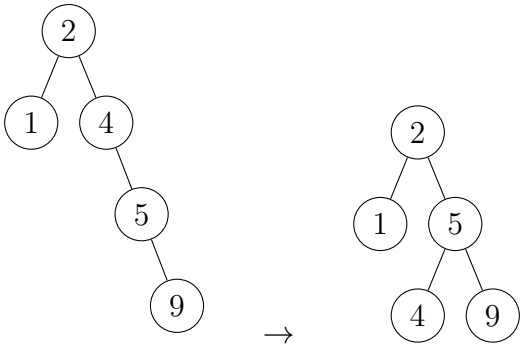
4:



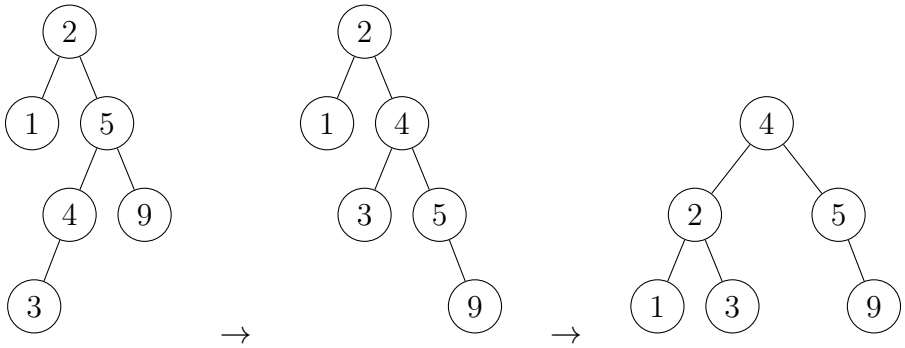
5:



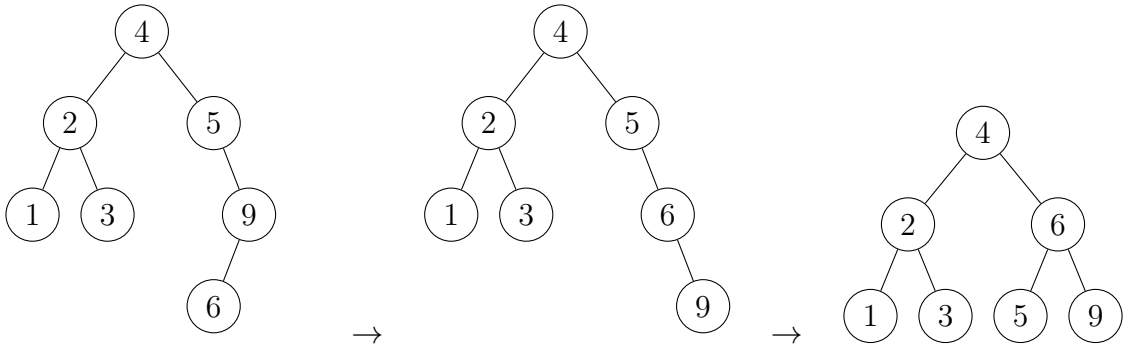
9: Single Rotation



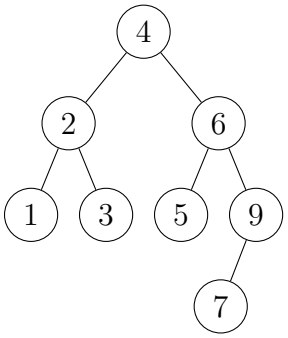
3: Double Rotation



6: Double Rotation



7:



(6). The following program, generating the output `'11 W World'`, shows the use of operators `size()`, `[]`, and `substr()` on the string class of the C++ standard library.

```
#include <iostream>
using namespace std;
int main(){
    string str="Hello World";
    cout << str.size()<<' '<< str[6]<<' '<<str.substr(6)<<'\n';
    return 0;
}
```

Complete the following recursive C++ function `reverse()` by using the operators `size()`, `[]`, and `substr()`, so that given a string of characters, `reverse()` prints characters of the string in reverse order. For instance, given `'hello', reverse()` prints `'olleh.'`

```
#include <iostream>
#include <string>
using namespace std;
void reverse(string str){
    cout << str[str.size()-1];
    if(str.size()<1)return;
    reverse(str.substr(0, str.size()-1));
}
int main(){
    string a = "hello world";
    reverse(a);
    return 0;
}
```