Laboratorio Nro. 1
Recursión

Andrés Julián Múnera
Universidad Eafit
Medellín, Colombia
ajmunerau@eafit.edu.co

Jorge Daniel Ruiz
Universidad Eafit
Medellín, Colombia
jdruizl@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1
3.2
3.3
3.5

```
public int factorial(int n) {
    if (n == 1) return 1;
  return n * factorial(n-1);
}
```

T(n): C1+T(n-1)
T(n) = C1 + C n
O(n)

```
public int fibonacci(int n) {
  if(n<2){
    return n;
    }else{
      return fibonacci(n-2)+fibonacci(n-1);
    }
  }
```

T(n) = C1 + T(n-2) + T(n - 1)

```
  public int sumDigits(int n) {
if(n < 10)
                 return n;
      return sumDigits(n/10) + n%10;
}
```

T(n)= C1 + T(n /10)
T(n) = c1 + $\boldsymbol{C\ log}$(n) $\boldsymbol{log(10)}$
O (log (n))

```
public int powerN(int base, int n) {
  if(n == 1){
    return base;
  }else{
    return base*powerN(base, n-1);
  }

  public String allStar(String str) {
      if(str.length() < 2)
      return str;
  return str.charAt(0) + "*" + allStar(str.substring(1));
}

public boolean groupSum6(int start, int[] nums, int target) {
  if(target == 0 && start == nums.length){
    return true;
  }
  if(start == nums.length){
    return false;
  }
  if(nums[start] != 6 && !groupSum6(start +1, nums, target - nums[start])){
    return groupSum6(start +1, nums, target);
  }
  else if(nums[start] == 6){
    return groupSum6(start +1, nums, target -nums[start]);
  }
  else{
    return true;
  }
T(n) = C + 2T(n-1)
T(n) = c1 2 ^ n-1 + C (2 ^ n – 1)
O(2 ^ n )

}

public boolean groupNoAdj(int start, int[] nums, int target) {
    if(target == 0 && start >= nums.length){
    return true;
  }
  if(start >= nums.length){
    return false;
  }
  if(!groupNoAdj(start +2, nums, target - nums[start])){
    return groupNoAdj(start +1, nums, target);
  }
  else{
    return true;
  }
}
```

T(n) = C + 2T(n-1)
T(n) = c1 2 ^n-1 + C (2 ^n – 1)
O(2 ^ n)

```
public boolean groupSum5(int start, int[] nums, int target) {
    if(target == 0 && start == nums.length){
    return true;
  }
  if(start >= nums.length){
    return false;
  }
  if(!(nums[start] % 5 == 0) && !groupSum5(start +1, nums, target - nums[start])){
    return groupSum5(start +1, nums, target);
  }
  else if((nums[start] % 5 == 0) && (start < nums.length - 1) && (nums[start +1] == 1)){
    return groupSum5(start +2, nums, target -nums[start]);
  }
  else if(nums[start] % 5 == 0){
    return groupSum5(start +1, nums, target -nums[start]);
  }
  else{
    return true;
  }
}
```

T(n) = C + 2T(n-1)
T(n) = c1 2 ^n-1 + C (2 ^n – 1)
O(2 ^ n)

```
public boolean s(int start, int[] nums, int a , int b) {
  if(start == nums.length){
    return a == b;
  }
  if(nums[start] %3 == 0){
    return s(start +1, nums, a + nums[start], b);
  }
  else if(nums[start] %5 == 0){
    return s(start +1, nums, a, b + nums[start]);
  }
  else if(!s(start +1, nums, a + nums[start], b)){
    return s(start +1, nums, a, b + nums[start]);
  }
  else{
    return true;
  }
}
public boolean split53(int [] nums){
  int a = 0;
  int b = 0;
  return s(0, nums, a, b);
}
```

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

T(n) = c1 2 ^n-1 + C (2 ^n – 1)

O(2 ^ n)

3.6

N: es la longitud del arreglo

Target: es un número entero al que deseas llegar

Start: es un número entero que va a incrementar para alcanzar cada posición

4) Simulacro de Parcial

4.1

False

s.substring(0, s.length()-1) == s.substring(s.length()-1, 0)

4.2 A

4.3

4.3.1 int res = solucionar(n-a,a,b,c) + 1;

4.3.2 res = Math.max(res, solucionar(n-c,a,b,c)+1));

4.3.3 res = Math.max(res, solucionar(n-b,a,b,c)+1));

4.5

4.5.1

If(T<=2){

Return 0;

}

Return ways(T-1) + ways(T-2)

4.5.2

B

4.9

C

4.10

B

4.11.1

return lucas(n – 1) + lucas(n – 2)

4.11.2

C

4.12

4.12.1 return sat;

4.12.2 return sat += Math.max(Fi, Fj);

4.12.3 return sat;