PUC Minas

Pontifícia Universidade Católica de Minas Gerais no São Gabriel
Curso de Sistemas de Informação
Algoritmos e Estruturas de Dados
Prof. João Caram - Semestre 2/2019
Exercícios - análise de complexidade de algoritmos - Valor: 5 pontos

Faça a análise dos códigos abaixo, incluindo melhor e pior caso, quando for necessário.

1)

```
public int q1(int[] vet, int pos){ //só funciona com vetor de tamanho ímpar
    if(pos == vet.Length/2) return vet[pos];
    else return (vet[pos] + vet[vet.Length-1-pos] + q1(vet, pos+1));
}
```

2)

```
int[] nãoRecursivo(int[] v, int[] w){
      int i = v.Length; int j = w.Length;
      int cont = 0;
      int[] novo = new int[i+j];
    while(cont<i && cont<j){
        novo[cont] = v[cont]+w[cont];
        cont++;
      }
    if(cont<i)
      for(int k=cont;k<i;k++) novo[k] = v[k];
    else
      for(int k=cont;k<j;k++) novo[k] = w[k];
}
```

3)

```
int[] alg3(int[] n, int[] m){  //m e n são do mesmo tamanho
      int[] sum = new int[n.Length];
    for(int i=0;i<n.Length;i++){
        sum[i] = n[i];
      for(int j=0; j<m.Length;j++)
          sum[i]+=m[j];
      }
      return sum;
}
```

4)

```
static int[] alg4(int[] arr1, int[] arr2){
      int i = 0;
      int j = 0;
      int[] arr3 = new int[arr1.Length+arr2.Length];
    for (i = 0; i < arr1.Length; i++)
      arr3[j++] = arr1[i];
    for (i = 0; i < arr2.Length; i++)
      arr3[j++] = arr2[i];
      return arr3;
}
```

5)

```
public void alg6(int[] intArray){
      int temp, j;
      for (int i = 1; i < intArray.Length; i++){
            temp = intArray[i];
            j = i - 1;
            while (j >= 0 && intArray[j] > temp){
                  intArray[j + 1] = intArray[j];
                  j--;
            }
            intArray[j + 1] = temp;
      }
}
```

6)

```
int algR(int[] v, int pos){
	if(pos==0) return v[0]*10;
	else return v[pos]*10 + algR(v, pos-1);
}
```

7)

```
int somaDigitos(int num){
	if(num==0)
		return 0;
	else{
		int aux = num%10;
		int n = num/10;
		return aux + somaDigitos(n);
	}
}
```

$S(0) = 1$

$S(n) = S(n/10) + 1$

$S(1) = S(1/0) + 1$

$= S(0) + 1$

$= 1 + 1$

$= 2$

5) • $S(1) = 1$ ← 2 operações

• $S(n) = 2 + S(n+1)$

$S(n+1) = 2 + S(n+2)$

$S(n+2) = 2 + S(n+3)$

substituir

$S(n) = 2 + (2 + (S\,n + 2))$

$= 4 + (S\,n + 2)$

$= 4 + (2\ S(n+3))$

$= 6 + S(n+3)$

$S(n) = 2i + S(n+i)$

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

$= 2 - 2n + 1$

$= 3 - 2n$

$n + i = 1$

$i = 1 - n$

6) • $S(0) = 1$

• $S(n) = 2 + S(n-1)$

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

$S(n-2) = 2 + S(n-3)$

$S(n) = 2 + (2 + S(n-2))$

$= 4 + S(n-2)$

$= 4 + (2 + S(n-3))$

$= 6 + (S(n-3))$

• $S(n) = 2i + S(n-i)$

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

$= 2(1+n) + S(0)$

$= 2 + 2n + 1$

$= 3 + 2n$

$n - i = 0$

$-i = -n$

$i = n$

7) • $S(0) = 1$

• $S(n) = S(n/10) + 1$

• $S(n/10) + 1 = S(n/100) + 1$

$S(n/100) + 2 = S(n/1000) + 1$

$S(n) = S(n/10^i) + i$ ⟶ $S(1) + \log_{10} n =$

$= \log_{10} n + 2$

hyp: $\frac{n}{10} = 1 \Rightarrow n = 10^i$

$\Rightarrow i = \log_{10} n$

2) melhor caso: while falso sempre

$$f(n) = 1 \cdot n - 1 \cdot 1$$
$$= n - 1$$

3) $f(n) = n \cdot (1 + n)$
$= n^2 + n$

4) $f(n) = (n+1) + (n+1)$
$= 2n + 2$

5) melhor caso = $f(n) = n - 1 \cdot 1$
$= n - 1$

pior caso = $f(n) = \frac{n^2 - n \cdot 2}{2}$

$= n^2 - n$