|  |  |  |  |
| --- | --- | --- | --- |
| **NMec:** | 93432 | **Nome:** | Joaquim Pedro Gonçalves Andrade Turma:P1 |

**Aula 3 – Análise da Complexidade de Algoritmos**

**1 -** Seja uma dada sequência (*array*) de n elementos inteiros. Pretende-se determinar quantos elementos da sequência são diferentes do seu elemento anterior. Ou seja:

array [i] ≠ array [i–1], para i > 0

* Implemente uma **função** **eficiente** e **eficaz** que determine quantos elementos (resultado da função) de uma sequência com n elementos (sendo n > 1) respeitam esta propriedade.

**Depois de validar o algoritmo apresente-o no verso da folha.**

* Determine experimentalmente a **ordem de complexidade do número de comparações** efetuadas pelo algoritmo e envolvendo elementos da sequência. Considere as seguintes 10 sequências de 10 elementos inteiros, todas diferentes, e que cobrem as distintas situações possíveis de execução do algoritmo. Determine, para cada uma delas, o número de elementos que obedecem à condição e o número de comparações efetuadas.

|  |  |  |
| --- | --- | --- |
| **Sequência** | **Resultado** | **N.º de operações** |
| {3, 3, 3, 3, 3, 3, 3, 3, 3, 3} | 0 | 9 |
| {4, 3, 3, 3, 3, 3, 3, 3, 3, 3} | 1 | 9 |
| {4, 5, 3, 3, 3, 3, 3, 3, 3, 3} | 2 | 9 |
| {4, 5, 1, 3, 3, 3, 3, 3, 3, 3} | 3 | 9 |
| {4, 5, 1, 2, 3, 3, 3, 3, 3, 3} | 4 | 9 |
| {4, 5, 1, 2, 6, 3, 3, 3, 3, 3} | 5 | 9 |
| {4, 5, 1, 2, 6, 8, 3, 3, 3, 3} | 6 | 9 |
| {4, 5, 1, 2, 6, 8, 7, 3, 3, 3} | 7 | 9 |
| {4, 5, 1, 2, 6, 8, 7, 9, 3, 3} | 8 | 9 |
| {4, 5, 1, 2, 6, 8, 7, 9, 3, 0} | 9 | 9 |

**Depois da execução do algoritmo responda às seguintes questões:**

* Em termos do número de comparações efetuadas, podemos distinguir alguma variação na execução do algoritmo? Ou seja, existe a situação de melhor caso e de pior caso, ou estamos perante um algoritmo com caso sistemático?

**Não podemos distinguir nenhuma variação, e, por isso, não existe pior, melhor caso ou caso médio. O algortimo é, então, sistemático.**

* Qual é a ordem de complexidade do algoritmo?

**A ordem de complexidade é linear. Ou seja, O(n).**

* Determine formalmente a ordem de complexidade do algoritmo. Tenha em atenção que deve obter uma expressão matemática exata e simplificada. **Faça a análise no verso da folha.**
* Calcule o valor da expressão para N = 10 e compare-o com os resultados obtidos experimentalmente.

**Apresentação do Algoritmo**

|  |
| --- |
| int property(int \*argv, char argc){  assert(argc>1);  int a=0, b=0;  for(int j=1;j<argc; j++){  b++;  if(argv[j]!=argv[j-1]){  a++;  }  }  return a;  } |

**Análise Formal do Algoritmo**

**E(n) = = N-1**

**N=10 ou seja 10-1=9**

**2 -** Seja uma dada sequência (*array*) de n elementos inteiros e não ordenada. Pretende-se determinar qual é o primeiro elemento da sequência que tem mais elementos menores do que ele atrás de si, e indicar a posição (índice do *array*) onde esse elemento se encontra.

Por exemplo, na sequência { 1, 9, 2, 8, 3, 4, 5, 3, 7, 2 } o elemento 7, que está na posição de índice 8 da sequência, é maior do que 6 elementos seus predecessores. Na sequência { 1, 7, 4, 6, 5, 2, 3, 2, 1, 0 } o elemento 6, que está na posição de índice 3 da sequência, é maior do que 2 elementos seus predecessores. Mas, na sequência { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 } nenhum elemento é maior do que qualquer um dos seus predecessores, pelo que deve ser devolvido -1 como resultado.

* Implemente uma **função** **eficiente** e **eficaz** que determine o índice do primeiro elemento (resultado da função) de uma sequência com n elementos (sendo n > 1) que tem o maior número de predecessores menores do que ele.

**Depois de validar o algoritmo apresente-o no verso da folha.**

* Determine experimentalmente a **ordem de complexidade do número de comparações** efetuadas envolvendo elementos da sequência. Considere as sequências anteriormente indicadas de 10 elementos inteiros e outras sequências diferentes à sua escolha. Determine, para cada uma delas, o índice do elemento procurado e o número de comparações efetuadas.

**Depois da execução do algoritmo responda às seguintes questões:**

* Em termos do número de comparações efetuadas, podemos distinguir alguma variação na execução do algoritmo? Ou seja, existe a situação de melhor caso e de pior caso, ou estamos perante um algoritmo com caso sistemático?

**Não existe nenhuma variação na execução do algoritmo, em termos de comparações, para o mesmo número de elementos. Não existe também melhor ou pior caso, sendo assim o algoritmo um caso sistemático.**

* Qual é a ordem de complexidade do algoritmo?

**A ordem de complexidade é O(n^2).**

* Determine formalmente a ordem de complexidade do algoritmo. Tenha em atenção que deve obter uma expressão matemática exata e simplificada. **Faça a análise no verso da folha.**
* Calcule o valor da expressão para N = 10 e compare-o com os resultados obtidos experimentalmente.

**Apresentação do Algoritmo**

|  |
| --- |
| int predecessores(int \*argv, int argc){  assert(argc>1);  int a=0, id=-1, b=0;  for(int i=1; i<argc; i++){  int c=0;  b++;  for(int j=(i-1); j>=0; j--){  if(argv[j]<argv[i]){  c++;  }  if (c>a){  a=c;  id=i;  }  }  }  return id;  } |

**Análise Formal do Algoritmo**

**E(n) ===(n(n-1))/2**

**E(10)= 45;**

**Os resultados práticos deram o mesmo resultado.**