Algoritmos e Estrutura de Dados II
(ESTCMP011)
2do PERIODO 2018

UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

Algoritmos de ordenação: selection & bubble sort.

Prof. Luis Cuevas Rodríguez, PhD
E-mail: lcuevasrodriguez@gmail.com / lrodriguez@uea.edu.br
Celular: 9298154648

EST
Escola Superior de Tecnologia da UEA

NUCOMP
NÚCLEO DE COMPUTAÇÃO

Conteúdo

- Especificado o algoritmo de ordenação por:
  - seleção (selection _sort)
  - flutuação (bubble_sort)
- Analisar seu tempo de execução.

Problema de ordenação

- Ordenar uma sequência de números em ordem crescente.:
  - Entrada: Uma sequência de *n* números $(a_1, a_2, ..., a_n)$ → chaves
  - Saída: Uma permutação (reordenação) $(a'_1, a'_2, . . . , a'_n)$ da sequência de entrada, tal que $a'_1 \leq a'_2 \leq . . . \leq a'_n$

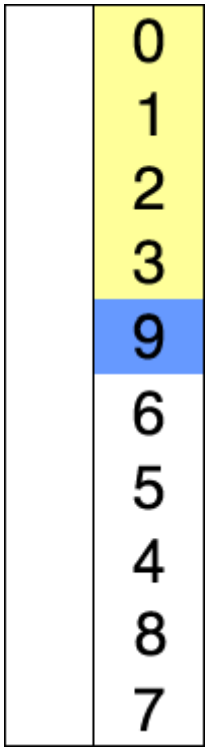
Entrada	Saída
(31,41,59,26,41,58)	(26,31,41,41,58,59)

- Instância do problema de ordenação →Uma sequência de entrada do problema que satisfaz as restrições impostas no enunciado do problema necessária para se calcular uma solução para o problema.

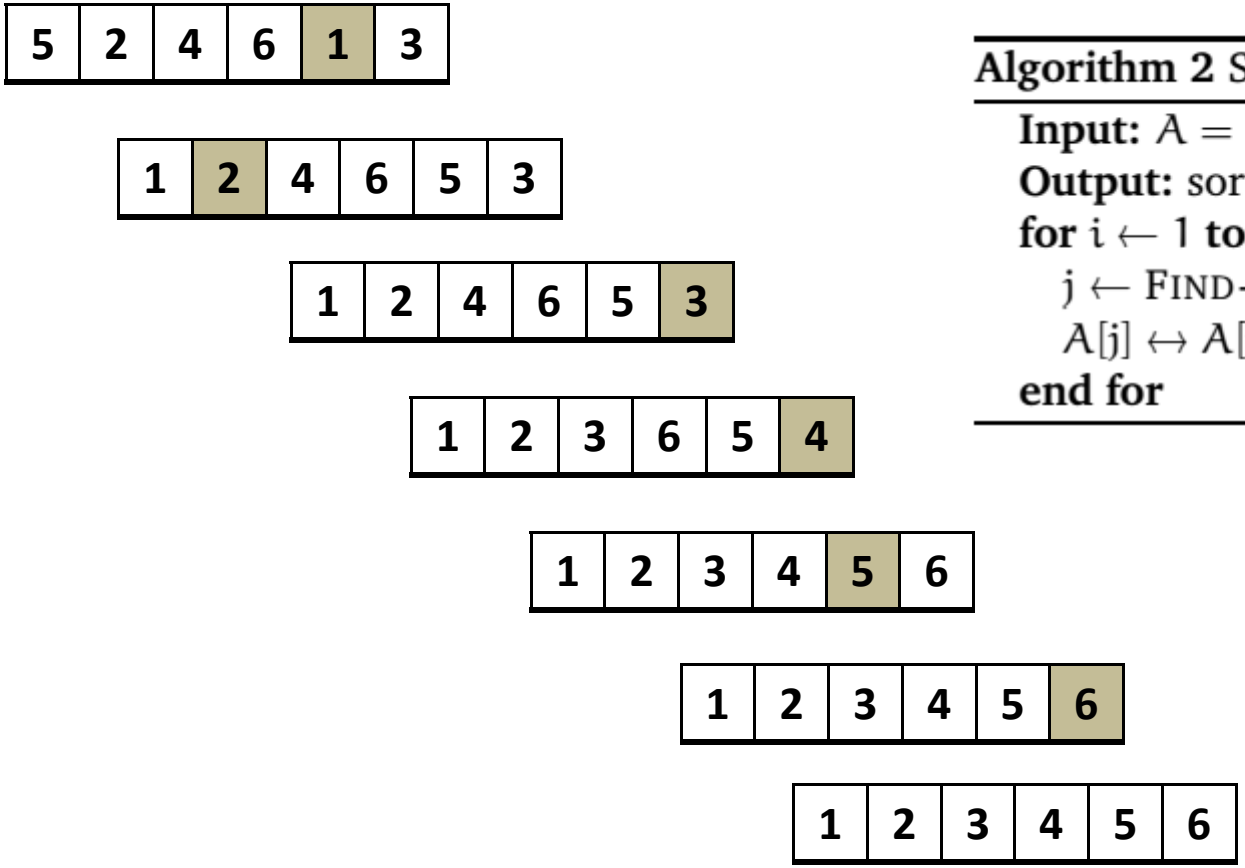
Ordenação por seleção (selection _sort)

- Ordenar de n números

1. Localizando primeiro o menor elemento do vetor,
2. permutar esse elemento como o elemento contido na primeira posição do vetor.
3. Em seguida encontrar o segundo menor elemento do vetor e trocar por o elemento na segunda posição.
4. Continuar para os primeiros n-1 elementos



Ordenação por seleção (selection _sort)



Algorithm 2 SELECTION-SORT(A)

Input: $A = \langle a_1, a_2, \ldots a_n \rangle$
Output: sorted A.
for $i \leftarrow 1$ **to** $n - 1$ **do**
  $j \leftarrow$ FIND-MIN(A, i, n)
  $A[j] \leftrightarrow A[i]$
end for

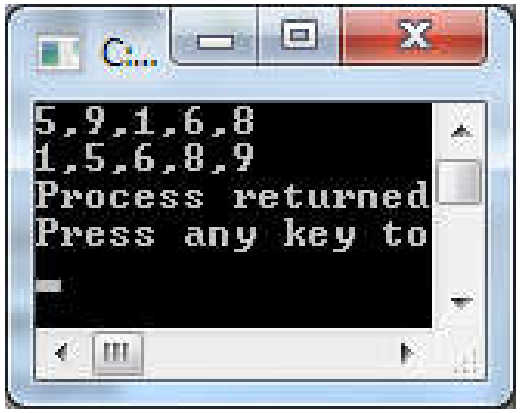
UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

Exercícios

- Ilustre a operação de SELECTION-SORT nos vetores
  - A = (31, 41, 59, 26, 41, 58).
  - B = (5, 1, 9, 6, 8).

Implementação - selection _sort

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    int temp, i, j, elem_min;
    imprime_vetor(lista);
    for (i=0; i<TAM-1; i++){
        elem_min = i;
        for (j = i+1; j < TAM; j++)
            if (lista[j]<lista[elem_min])
                elem_min = j;
        temp = lista[elem_min];
        lista[elem_min]=lista[i];
        lista[i] = temp;
    }
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```



Implementação - selection _sort

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    imprime_vetor(lista);
    ordena_selecao(lista);
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

```
void ordena_selecao(int *l){
    int i,j, elem_min, temp;
    for (i=0; i<TAM-1; i++){
        elem_min = i;
        for (j = i+1; j < TAM; j++)
            if (*(l+j)<*(l+elem_min))
                elem_min = j;
        temp = *(l+elem_min);
        *(l+elem_min)=*(l+i);
        *(l+i) = temp;
    }
}
```

Análise do algoritmo

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    int temp, i, j, elem_min;
    imprime_vetor(lista);
    for (i=0; i<TAM-1; i++){
        elem_min = i;
        for (j = i+1; j < TAM; j++)
            if (lista[j]<lista[elem_min])
                elem_min = j;
        temp = lista[elem_min];
        lista[elem_min]=lista[i];
        lista[i] = temp;
    }
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

c_1 1

c_4 n-1

c_6 n-j

Melhor caso: *?*

Pior caso: *?*

Análise do algoritmo

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    int temp, i, j, elem_min;
    imprime_vetor(lista);
    for (i=0; i<TAM-1; i++){
        elem_min = i;
        for (j = i+1; j < TAM; j++)
            if (lista[j]<lista[elem_min])
                elem_min = j;
        temp = lista[elem_min];
        lista[elem_min]=lista[i];
        lista[i] = temp;
    }
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

c_1 1

c_4 n-1

c_6 n-j

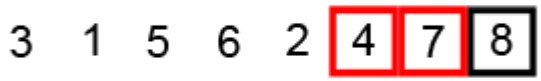
Melhor caso: $an+b = \Theta(n^2)$

Pior caso: $an^2+n+b = \Theta(n^2)$

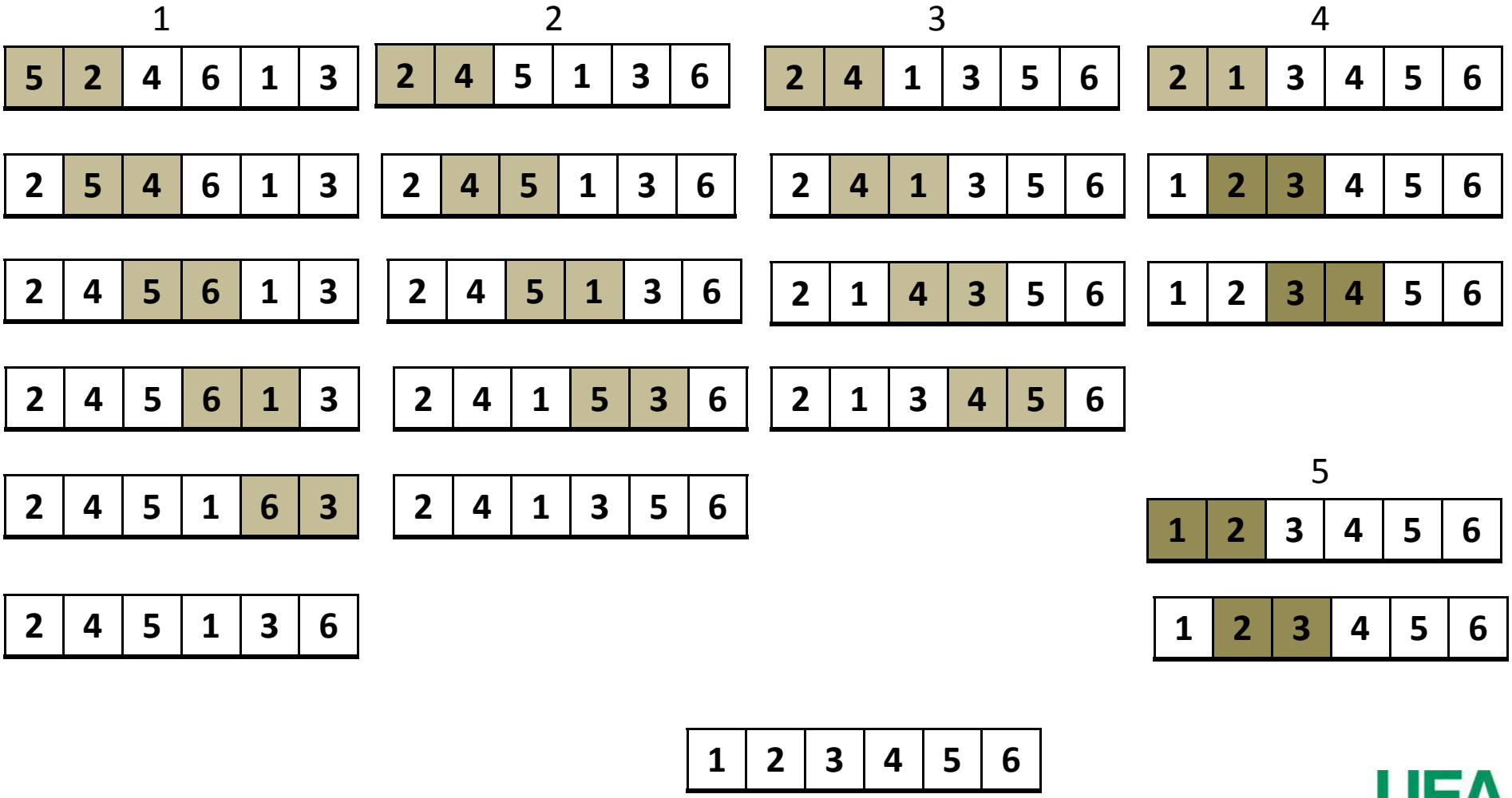
UEA
UNIVERSIDADE
DO ESTADO DO
AMAZONAS

Ordenação por flutuação (bubble_sort)

- Percorrer o vetor diversas vezes, e a cada passagem fazer flutuar para o final o maior elemento da sequência.
- Cada elemento da posição i será comparado com o elemento da posição i + 1. Se i > i+1 então trocam de lugar.



Ordenação por flutuação (bubble_sort)



Prof. Luis Cuevas Rodríguez, PhD

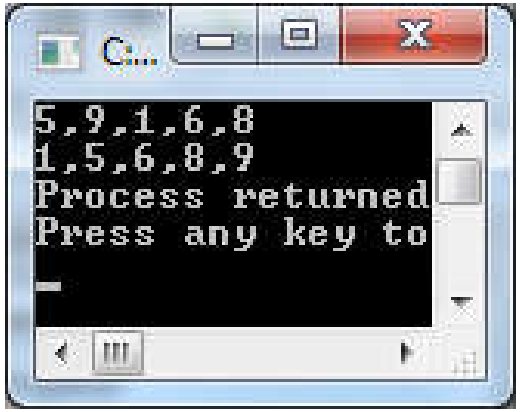
UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

Exercícios

- Ilustre a operação de BUBBLE-SORT nos vetores
  - A = (31, 41, 59, 26, 41, 58).
  - B = (5, 1, 9, 6, 8).

Implementação - bubble_sort

```
int main()
{
     int lista[TAM]={5,9,1,6,8};
     int temp, i, j;
     imprime_vetor(lista);
     for (i=0; i<TAM-1; i++){
         for (j = 0; j < TAM-i-1; j++)
             if (lista[j]>lista[j+1]){
                 temp = lista[j];
                 lista[j]=lista[j+1];
                 lista[j+1] = temp;
             }
     }
     cout << endl;
     imprime_vetor(lista);
     return 0;
}
```



Implementação - bubble_sort

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    imprime_vetor(lista);
    ordena_flutuacao(lista);
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

```
void ordena_flutuacao(int *l){
    int i,j,temp;
    for (i=0; i<TAM-1; i++){
        for (j = 0; j < TAM-i-1; j++)
            if (*(l+j)>*(l+j+1)){
                temp = *(l+j);
                *(l+j)=*(l+j+1);
                *(l+j+1) = temp;
            }
    }
}
```

Análise do algoritmo

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    int temp, i, j;
    imprime_vetor(lista);
    for (i=0; i<TAM-1; i++){
        for (j = 0; j < TAM-i-1; j++)
            if (lista[j]>lista[j+1]){
                temp = lista[j];
                lista[j]=lista[j+1];
                lista[j+1] = temp;
            }
    }
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

c_1 — 1

c_4 — n-1

c_5 — n-(i+1)

Melhor caso: *?*

Pior caso: *?*

Análise do algoritmo

```
int main()
{
    int lista[TAM]={5,9,1,6,8};
    int temp, i, j;
    imprime_vetor(lista);
    for (i=0; i<TAM-1; i++){
        for (j = 0; j < TAM-i-1; j++)
            if (lista[j]>lista[j+1]){
                temp = lista[j];
                lista[j]=lista[j+1];
                lista[j+1] = temp;
            }
    }
    cout << endl;
    imprime_vetor(lista);
    return 0;
}
```

c_1 — 1

c_4 — n-1

c_5 — n-(i+1)

Melhor caso: $an+b = \Theta(n^2)$

Pior caso: $an^2+n+b = \Theta(n^2)$

UEA
UNIVERSIDADE
DO ESTADO DO
AMAZONAS

Comparativas

- lista ordenada em ordem crescente.

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	0,0988	5050	0
Selection Sort	0,0602	4950	297
Insertion sort	0,0038	99	198

Tamanho do vetor= 100

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	9,5415	500500	0
Selection Sort	5,4587	499500	2997
Insertion sort	0,0359	999	1998

Tamanho do vetor= 1000

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	934,5364	50005000	0
Selection Sort	508,5891	49995000	29997
Insertion sort	0,3558	9999	19998

Tamanho do vetor= 10000

P

https://www.devmedia.com.br/algoritmos-de-ordenacao-analise-e-comparacao/28261

UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

Comparativas

- lista ordenada em ordem decrescente

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	0,2045	5050	14850
Selection Sort	0,0750	4950	297
Insertion sort	0,1173	99	5148

Tamanho do vetor= 100

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	20,3377	500500	1498500
Selection Sort	6,9038	499500	2997
Insertion sort	11,4277	999	501498

Tamanho do vetor= 1000

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	1838,0272	50005000	149985000
Selection Sort	665,2050	49995000	29997
Insertion sort	1074,1171	9999	50014998

Tamanho do vetor= 10000

https://www.devmedia.com.br/algoritmos-de-ordenacao-analise-e-comparacao/28261

Comparativas

- lista desordenada com números aleatórios

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	0,1596	5050	6777
Selection Sort	0,0698	4950	297
Insertion sort	0,0570	99	2457

Tamanho do vetor= 100

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	16,6730	500500	756840
Selection Sort	5,6664	499500	2997
Insertion sort	5,7523	999	254278

Tamanho do vetor= 1000

Algoritmo	Tempo(ms)	Comparações	Movimentações
Bubble sort	1455,9734	50005000	74237889
Selection Sort	545,1068	49995000	29997
Insertion sort	539,6891	9999	24765961

Tamanho do vetor= 10000

https://www.devmedia.com.br/algoritmos-de-ordenacao-analise-e-comparacao/28261

Bibliografía

- CORMEN, T.H., LEISERSON, C.E., RIVEST, R.L., STEIN, C. Algoritmos: Teoria e Prática. Tradução da 3a. edição. Rio de Janeiro: Elsevier, 2012.
- SZWARCFITER, J, L., MARKENZON, L. Estruturas de Dados e seus Algoritmos. 2a edição. Rio de Janeiro: LTC, 2010.
- ZIVIANI, N. Projeto de Algoritmos com Implementação em Pascal e C. 3a edição. São Paulo: Cengage Learning, 2010.
- STROUSTRUP, B. Programming: Principles and Practice Using C++. 2nd Edition. Addison-Wesley Professional. 2014.
- B. Stroustrup. The C++ Programming Language, 4th Edition, 2013.
- Feofiloff, P. Algoritmos em Linguagem C. Elsevier. 2009.
- AHO, A. V. et al. Data Structure and Algorithms. Readings, Addison-Wesley.
- WIRTH, N. Algoritmos e Estruturas de Dados. Rio de Janeiro: Ed. Prentice Hall do Brasil.
- KNUTH, D. E. The Art of Computer Programming. Vol. 1, Addison-Wesley, Reading, Mass.