Lista 6: ordenação

Prof. Felipe Duque
felipe.duque@ufpe.br

1. (5 pontos) Considere o vetor abaixo:

   2 3 5 7 13 11 19 17

   (a) Mostre etapas do funcionamento da ordenação por **seleção**. Quantas comparações e quantas trocas seriam necessárias para ordenar o vetor com esse algoritmo?

   (b) Mostre etapas do funcionamento da ordenação por **inserção**. Quantas comparações e quantas trocas seriam necessárias para ordenar o vetor com esse algoritmo?

2. (10 pontos) Considere o programa abaixo.

```
1#include <stdio.h>
2#include <stdlib.h>
3#include <string.h>
4#include <time.h>
5#define N 100
6void encontra_menor_indice(const int* vec, int i_ini, \
7            int n, int* q_comp, int* m_ind);
8void insercao(const int* vec, int n, int* vec_out, \
9            int* q_comp, int* q_troc);
10void selecao(const int* vec, int n, int* vec_out, \
11            int* q_comp, int* q_troc)
12{
13  int m_ind;
14  int total_comp = 0;
15  memcpy(vec_out, vec, n * sizeof(int));
16  for (int i = 0; i < n; i++)
```

```
17   {
18     encontra_menor_indice(vec_out, i, n, q_comp, &m_ind);
19     int tmp = vec_out[m_ind];
20     vec_out[m_ind] = vec_out[i];
21     vec_out[i] = tmp;
22
23     total_comp += (*q_comp);
24     (*q_troc)++;
25   }
26 *q_comp = total_comp;
27 }
28
29 int main()
30 {
31   int i;
32   int vec[N], vec_ins[N], vec_sel[N];
33   int q_comp_sel = 0, q_troc_sel = 0;
34   int q_comp_ins = 0, q_troc_ins = 0;
35
36   srand(time(NULL));
37
38   for (i = 0; i < N; i++)
39     vec[i] = rand() % (10*N);
40
41   selecao(vec, N, vec_sel, &q_comp_sel, &q_troc_sel);
42   insercao(vec, N, vec_ins, &q_comp_ins, &q_troc_ins);
43
44   printf("Selecao:\n");
45   printf("Numero de comparacoes: %d\n", q_comp_sel);
46   printf("Numero de trocas: %d\n", q_troc_sel);
47   for (i = 0; i < N; i++)
48     printf("%d ", vec_sel[i]);
49
50   printf("\n");
51   printf("\n");
52
53   printf("Insercao:\n");
54   printf("Numero de comparacoes: %d\n", q_comp_ins);
```

```
55   printf("Numero de trocas: %d\n", q_troc_ins);
56   for (i = 0; i < N; i++)
57     printf("%d ", vec_ins[i]);
58   printf("\n");
59   return 0;
60 }
```

(a) Explique o objetivo e o funcionamento do programa.

(b) Implemente a função `encontra_menor_indice()` com a seguinte assinatura:

```
1 void encontra_menor_indice(const int* vec, int i_ini,\
2                  int n, int* q_comp, int* m_ind)
```

que deverá encontrar o índice `m_ind` do menor elemento no vetor de inteiros `vec` de tamanho `n` a partir do índice `i_ini`. Além disso, em `q_comp`, a função deverá armazenar a quantidade de comparações realizadas na função.

(c) Implemente a função `insercao()` com a seguinte assinatura:

```
1 void insercao(const int* vec, int n, int* vec_out, \
2              int* q_comp, int* q_troc)
```

Ela deverá ordenar, com o algoritmo de inserção, o vetor `vec` de tamanho `n`, preencher `vec_out` com o vetor ordenado, `q_comp` com a quantidade de comparações realizadas e `q_troc` com a quantidade de trocas realizadas.

(d) Compile e execute o programa inteiro e compare os resultados obtidos.