Universidade Federal de São João del Rei
Departamento de Ciência da Computação
Curso de Ciência da Computação

Roteiro 10

Rodrigo José Zonzin
212050002

1 Ordenacao e ordenação invertida

Todos os algoritmos e suas variações estão implementados no codigo a seguir.

```c
#include <stdio.h>
#include <stdlib.h>


void insertionSort(int *vetor, int N){
    int i, chave, j;
    for(i = 1; i < N; i++){
        chave = vetor[i];
        j = i - 1;

        while(j >= 0 && vetor[j] > chave){
            vetor[j + 1] = vetor[j];
            j = j - 1;
        }
        vetor[j + 1] = chave;
    }
}

void insertionSortReverse(int *vetor, int N){
    int i, chave, j;
    for(i = 1; i < N; i++){
        chave = vetor[i];
        j = i - 1;

        while(j >= 0 && vetor[j] < chave){
            vetor[j + 1] = vetor[j];
            j = j - 1;
        }
        vetor[j + 1] = chave;
    }
}

void selectionSort(int *vetor, int N){
    int i, j, min_idx;

    for(i = 0; i < N - 1; i++){
        min_idx = i;
        for(j = i + 1; j < N; j++){
            if(vetor[j] < vetor[min_idx]){
```

```
40                  min_idx = j;
41              }
42          }
43
44          int temp = vetor[min_idx];
45          vetor[min_idx] = vetor[i];
46          vetor[i] = temp;
47      }
48  }
49
50  void selectionSortReverse(int *vetor, int N){
51      int i, j, max_idx;
52
53      for(i = 0; i < N - 1; i++){
54          max_idx = i;
55          for(j = i + 1; j < N; j++){
56              if(vetor[j] > vetor[max_idx]){
57                  max_idx = j;
58              }
59          }
60
61          int temp = vetor[max_idx];
62          vetor[max_idx] = vetor[i];
63          vetor[i] = temp;
64      }
65  }
66
67  void bubbleSort(int vetor[], int N){
68      int i, j;
69      for(i = 0; i < N - 1; i++){
70          for(j = 0; j < N - i - 1; j++){
71              if(vetor[j] > vetor[j + 1]){
72                  int temp = vetor[j];
73                  vetor[j] = vetor[j + 1];
74                  vetor[j + 1] = temp;
75              }
76          }
77      }
78  }
79
80  void bubbleSortReverse(int vetor[], int N){
81      int i, j;
82      for(i = 0; i < N - 1; i++){
83          for(j = 0; j < N - i - 1; j++){
84              if(vetor[j] < vetor[j + 1]){
85                  int temp = vetor[j];
86                  vetor[j] = vetor[j + 1];
87                  vetor[j + 1] = temp;
88              }
89          }
90      }
91  }
92
93
94  int main(int argc, char** argv){
95      int N = 0;
96      fscanf(stdin, "%d", &N);
97      int *vet  =(int*)malloc(sizeof(int)*N);
98
```

```
 99      for(int i =0; i<N; i++){
100          fscanf(stdin, "%d", &vet[i]);
101      }
102
103      int alg = atoi(argv[1]);
104
105      if(alg == 1){
106          bubbleSort(vet, N);
107          for(int i =0; i<N; i++){
108              fprintf(stdout, "%d  ", vet[i]);
109          }
110          printf("\n");
111      }
112
113      if(alg == 12){
114          bubbleSortReverse(vet, N);
115          for(int i =0; i<N; i++){
116              fprintf(stdout, "%d  ", vet[i]);
117          }
118          printf("\n");
119      }
120
121      if(alg == 2){
122          selectionSort(vet, N);
123          for(int i =0; i<N; i++){
124              fprintf(stdout, "%d  ", vet[i]);
125          }
126          printf("\n");
127      }
128
129      if(alg == 22){
130          selectionSortReverse(vet, N);
131          for(int i =0; i<N; i++){
132              fprintf(stdout, "%d  ", vet[i]);
133          }
134          printf("\n");
135      }
136
137      if(alg == 3){
138          insertionSort(vet, N);
139          for(int i =0; i<N; i++){
140              fprintf(stdout, "%d  ", vet[i]);
141          }
142          printf("\n");
143      }
144
145      if(alg == 32){
146          insertionSortReverse(vet, N);
147          for(int i =0; i<N; i++){
148              fprintf(stdout, "%d  ", vet[i]);
149          }
150          printf("\n");
151      }
152
153      return 0;
154  }
```

../ex11/ordenacao.c

```
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 1 < teste.txt
1  3  5  8  9  12  25  32  35  36  54  65  69  74  74  82  84  85  87  98
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 12 < teste.txt
98  87  85  84  82  74  74  69  65  54  36  35  32  25  12  9  8  5  3  1
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ echo Bubble Sort
Bubble Sort
```

Figura 1: Bubble Sort

```
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 2 < teste.txt
1  3  5  8  9  12  25  32  35  36  54  65  69  74  74  82  84  85  87  98
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 22 < teste.txt
98  87  85  84  82  74  74  69  65  54  36  35  32  25  12  9  8  5  3  1
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ echo Selection Sort
Selection Sort
```

Figura 2: Selection Sort

```
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 3 < teste.txt
1  3  5  8  9  12  25  32  35  36  54  65  69  74  74  82  84  85  87  98
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ ./ord 32 < teste.txt
98  87  85  84  82  74  74  69  65  54  36  35  32  25  12  9  8  5  3  1
zonzin@rodrigo:~/Documentos/Faculdade/labaedsII/labaeds2/pr10/ex11$ echo Insertion Sort
Insertion Sort
```

Figura 3: Insertion Sort

2 Complexidade - deu errado

Por algum motivo, o comportamento assintótico não está sendo observado em nenhum dos plots. Em razão do horário (23h37), eu desisto.

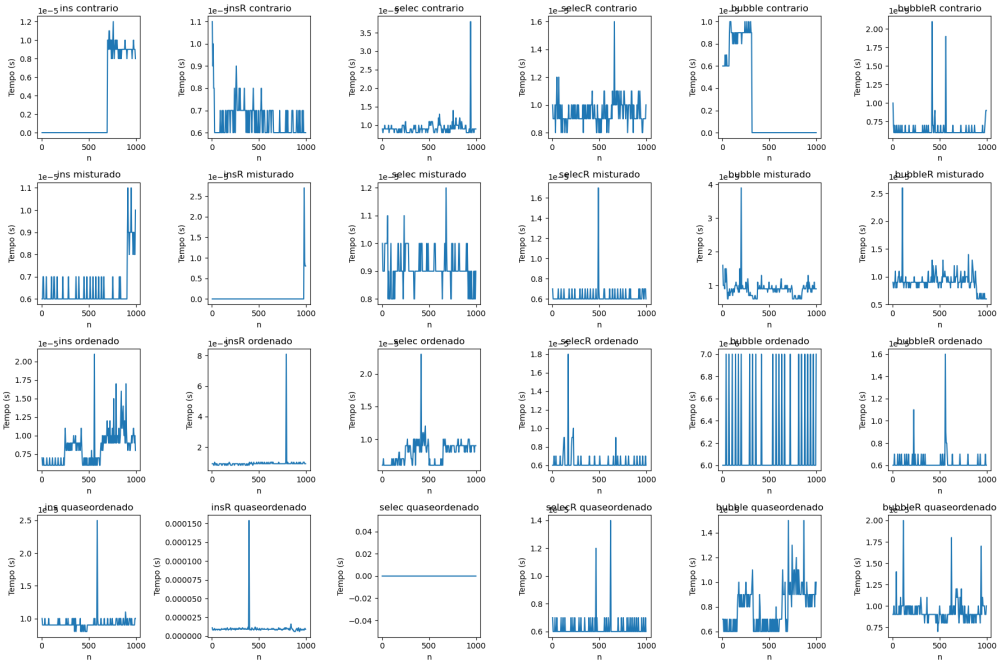

Figura 4: Complexidade - Erro

3 Struct

struct Pessoa int idade; char nome[30];

Sejam Pessoas p1 e p2, temos

```
if(p1->idade >= p2->idade)
  doSomething();

else{
  doOtherThing();
}
```

```
  if(strcmp(p1->nome, p2->nome) < 0)
    doSomething();

  else if(strcmp(p1->nome, p2->nome)) > 0){
    doOtherThing();
  }
```