# RELATÓRIO – PROJETO 5 (ORDENAÇÃO)

**CASO DE TESTE 02**

Caso de teste com o vetor de 120.000 posições e o vetor de busca com 240.000 posições. Com isso, obtemos a média dos resultados de cada iteração com o menu:

OP1: Busca sem ordenação

Tempo ordenação: 24.225316

OP2: Busca sequencial com vetor ordenado por insertion sort

Tempo ordenação: 7.3167628

Tempo busca: 10.396088

Tempo total: 17.712851

OP3: Busca sequencial com vetor ordenado por shell sort

Tempo ordenação: 0.027888

Tempo busca: 10.838090

Tempo total: 10.865977

OP4: Busca sequencial com vetor ordenado por quick sort

Tempo ordenação: 0.015338

Tempo busca: 10.519819

Tempo total: 10.535156

OP5: Busca binária com vetor ordenado por insertion sort

Tempo ordenação: 7.498058

Tempo busca: 2.822885

Tempo total: 10.320943

OP6: Busca binária com vetor ordenado por shell sort

Tempo ordenação: 0.021613

Tempo busca: 2.871193

Tempo total: 2.892806

OP7: Busca binária com vetor ordenado por quick sort

Tempo ordenação: 0.015358

Tempo busca: 2.874333

Tempo total: 2.889691

De fato, após realizar todos esses testes e comparações, podemos afirmar que a busca binária certamente é mais vantajosa e mais rápida do que a busca sequencial. Além disso, vale ressaltar que, nos casos estudados, os métodos shell sort e quick sort se saíram muito melhores do que o método de ordenação por insertion.

**CASO DE TESTE 03**

Caso de teste com o vetor de 240.000 posições e o vetor de busca com 480.000 posições. Com isso, obtemos a média dos resultados de cada iteração com o menu:

OP1: Busca sem ordenação

Tempo ordenação: 89.643492

OP2: Busca sequencial com vetor ordenado por insertion sort

Tempo ordenação: 31.234610

Tempo busca: 38.767966

Tempo total: 70.002577

OP3: Busca sequencial com vetor ordenado por shell sort

Tempo ordenação: 0.045019

Tempo busca: 42.992245

Tempo total: 43.037264

OP4: Busca sequencial com vetor ordenado por quick sort

Tempo ordenação: 0.030497

Tempo busca: 38.819530

Tempo total: 38.850027

OP5: Busca binária com vetor ordenado por insertion sort

Tempo ordenação: 31.338310

Tempo busca: 6.601902

Tempo total: 37.940212

OP6: Busca binária com vetor ordenado por shell sort

Tempo ordenação: 0.046812

Tempo busca: 6.803732

Tempo total: 6.850544

OP7: Busca binária com vetor ordenado por quick sort

Tempo ordenação: 0.027940

Tempo busca: 6.785896

Tempo total: 6.813835

**CASO DE TESTE 04**

Caso de teste com o vetor de 480.000 posições e o vetor de busca com 960.000 posições. Com isso, obtemos a média dos resultados de cada iteração com o menu:

OP1: Busca sem ordenação

Tempo ordenação: 359.179671

OP2: Busca sequencial com vetor ordenado por insertion sort

Tempo ordenação: 122.121383

Tempo busca: 143.929933

Tempo total: 266.051315

OP3: Busca sequencial com vetor ordenado por shell sort

Tempo ordenação: 0.092364

Tempo busca: 152.620174

Tempo total: 152.712538

OP4: Busca sequencial com vetor ordenado por quick sort

Tempo ordenação: 0.061410

Tempo busca: 136.081299

Tempo total: 136.142709

OP5: Busca binária com vetor ordenado por insertion sort

Tempo ordenação: 124.611435

Tempo busca: 14.940395

Tempo total: 139.551830

OP6: Busca binária com vetor ordenado por shell sort

Tempo ordenação: 0.106093

Tempo busca: 14.229547

Tempo total: 14.335640

OP7: Busca binária com vetor ordenado por quick sort

Tempo ordenação:

Tempo busca:

Tempo total:

Neste caso, como estamos tratando de um vetor de 960.000 elementos, podemos concluir que a busca por ordenação, assim como em todos os outros casos, foi a que apresentou pior desempenho. Nessa lógica, analisamos que, mesmo com a grande quantia de elementos, a busca binária consegue ser relativamente rápida. Por fim, assim como observado nos outros casos de teste, os algoritmos de ordenação por shell sort e por quick sort se saíram muito melhores se comparados ao método de ordenação por insertion sort.