REPORT BIG ASSIGNMENT
Chương 1

Họ tên: Nguyễn Anh Tuấn
MSSV: 20200400

Requirement:

Viết chương trình mô phỏng trực quan các thuật toán sắp xếp. Yêu cầu xuất ra được độ phức tạp, thời gian chạy của mỗi thuật toán
Các thuật toán gồm:
1. Quick_sort
2. Merge_sort
3. Natural_merge_sort
4. Heap_sort
5. Radix_sort

Bài làm:

Code:

Các thuật toán sắp xếp theo yêu cầu:

- QuickSort

```
245   void QuickSort(int a[], int l, int r){
246       int i, j;
247       if (l >= r) return;
248       int x = a[(l+r)/2];
249       i = l; j = r;
250        do{
251           while(a[i] < x) i++;
252           while(a[j] > x) j--;
253           if(i <= j) {
254               swap(a[i], a[j]);
255           i++ ; j--;
256           }
257       } while(i < j) ;
258       if(l<j) QuickSort(a, l, j);
259       if(i<r) QuickSort(a, i, r);
260   }
```

- HeapSort

```
299   void heapify(int a[], int n, int start){
300       int l = start*2+1;
301       int r = start*2+2;
302       int large = start;
303       if ( l < n && a[l] > a[large]) large = l;
304       if ( r < n && a[r] > a[large]) large = r;
305       if (large != start){
306           swap(a[large], a[start]);
307           heapify(a,n,large);
308       }
309   }
310
311 > /**…
317
318   void heapSort( int a[], int n){
319       for ( int i = n/2-1; i >= 0; i--){
320           heapify(a,n,i);
321       }
322       for ( int i = n-1; i >= 0; i--){
323           swap(a[i], a[0]);
324           heapify(a,i,0);
325       }
326   }
```

- MergeSort

```
61  void merge(int a[], int o, int m, int r) {
62      int i, j, k;
63      int n1 = m - o + 1;
64      int n2 = r - m;
65      int L[n1], R[n2];
66      for (i = 0; i < n1; i++) L[i] = a[o + i];
67      for (j = 0; j < n2; j++) R[j] = a[m + 1 + j];
68      i = 0; j = 0; k = o;
69      while (i < n1 && j < n2) {
70          if (L[i] <= R[j]) {
71              a[k] = L[i];
72              i++;
73          } else {
74              a[k] = R[j];
75              j++;
76          }
77          k++;
78      }
79      while (i < n1) {
80          a[k] = L[i];
81          i++; k++;
82      }
83      while (j < n2) {
84          a[k] = R[j];
85          j++; k++;
86      }
87  }
88
89  > /**…
95
96  void mergeSort(int a[], int n) {
97      int size;
98      int start;
99      for (size = 1; size <= n-1; size = 2*size) {
100         for (start = 0; start < n-1; start += 2*size) {
101             int m = min(start + size - 1, n-1);
102             int re = min(start + 2*size - 1, n-1);
103             merge(a, start, m, re);
104         }
105     }
106 }
```

- Natural Merge Sort

```
177  void naturalMergeSort(int a[], int n) {
178      bool check = true;
179      int l = 0, r = 0, m = 0;
180      cout << "Thu tu mang moi lan merge: \n";
181      while (check) {
182          check = false;
183          l = 0;
184          while (l < n) {
185              r = l + 1;
186              while (r < n && a[r] >= a[r-1]) {
187                  r++;
188              }
189              if (r < n) {
190                  m = r + 1;
191                  while (m < n && a[m] >= a[m-1]) {
192                      m++;
193                  }
194                  merge(a, l, r - 1, m - 1);
195                  l = m;
196                  check = true;
197              } else {
198                  l = r;
199              }
200          }
201      }
202  }
```

- Radix Sort

```
379  int getMax(int a[], int n)
380  {
381      int max = a[0];
382      for (int i = 1; i < n; i++){
383          if (a[i] > max) max = a[i];
384      }
385      return max;
386  }
387
388  /**…
395
396  void count(int a[], int n, int exp)
397  {
398      int output[n];
399      int i, count[10] = { 0 };
400      for (i = 0; i < n; i++) count[(a[i] / exp) % 10]++;
401      for (i = 1; i < 10; i++) count[i] += count[i - 1];
402      for (i = n - 1; i >= 0; i--) {
403          output[count[(a[i] / exp) % 10] - 1] = a[i];
404          count[(a[i] / exp) % 10]--;
405      }
406      for (i = 0; i < n; i++) a[i] = output[i];
407  }
408
409  /**…
415
416  void radixSort(int a[], int n) {
417      int m = getMax(a, n);
418      for (int i = 1; m / i > 0; i *= 10) count(a, n, i);
419  }
```

Các hàm bổ trợ:

```
int a[] = {1,5,2,3,25,1,3,5,6,34,2,3,7,345,123,5,42};
int b[] = {1,5,2,3,25,1,3,5,6,34,2,3,7,345,123,5,42};
int n = 17;

/**...

void log(int a[], int n){
    for ( int i = 0; i < n; i++ ){
        cout << a[i] << " ";
    }
    cout << endl;
}

/**...

void swap(int &a,int &b){
    int tmp = a;
    a = b;
    b = tmp;
}
```

Kết quả:

```cpp
int main(){
    int sort;
    cout << "-----------------------------------------\n";
    cout << "(1) QuickSort \n";
    cout << "(2) MergeSort \n";
    cout << "(3) NaturalMergeSort \n";
    cout << "(4) HeapSort \n";
    cout << "(5) RadixSort \n";
    cout << "-----------------------------------------\n";
    cout << "Chon thuat toan muon dung de sap xep mang: ";
    cin >> sort;
    cout << "-----------------------------------------\n";
    while ( sort >= 6 || sort <= 0){
        cout << "Vui long nhap lai option tu 1 -> 5: ";
        cin >> sort;
        cout << "-----------------------------------------\n";
    }
    clock_t start = clock();

    switch (sort){
        case 1: {
            cout << "Thuat toan Quick Sort: \n";
            cout << "Mang truoc khi sap xep: "; log(a,n);
            QuickSort(a,0,n-1);
            int time = clock() - start;
            cout << "Thu tu mang moi lan swap vi tri pivot: \n";
            QuickSortWithLog(b,0,n-1);
            cout << "Mang sau khi sap xep: ";
            log(a,n);
            cout << "Time complex: " << time << " ms\n";
            cout << "-----------------------------------------\n";
            break;
        }
        case 2: {
            cout << "Thuat toan Merge Sort: \n";
            cout << "Mang truoc khi sap xep: "; log(a,n);
            mergeSort(a,n);
            int time = clock() - start;
            mergeSortWithLog(b,n);
            cout << "Mang sau khi sap xep: ";
            log(a,n);
            cout << "Time complex: " << time << " ms\n";
            cout << "-----------------------------------------\n";
            break;
        }
        case 3: {
            cout << "Thuat toan Natural Merge Sort: \n";
            cout << "Mang truoc khi sap xep: "; log(a,n);
            naturalMergeSort(a,n);
            int time = clock() - start;
            naturalMergeSortWithLog(b,n);
            cout << "Mang sau khi sap xep: ";
            log(a,n);
            cout << "Time complex: " << time << " ms\n";
            cout << "-----------------------------------------\n";
            break;
        }
        case 4: {
            cout << "Thuat toan Heap Sort: \n";
            cout << "Mang truoc khi sap xep: "; log(a,n);
            heapSort(a,n);
            int time = clock() - start;
            heapSortWithLog(b,n);
            cout << "Mang sau khi sap xep: ";
            log(a,n);
            cout << "Time complex: " << time << " ms\n";
            cout << "-----------------------------------------\n";
            break;
        }
        case 5: {
            cout << "Thuat toan Radix Sort: \n";
            cout << "Mang truoc khi sap xep: "; log(a,n);
            radixSort(a,n);
            int time = clock() - start;
            radixSortWithLog(b,n);
            cout << "Mang sau khi sap xep: ";
            log(a,n);
            cout << "Time complex: " << time << " ms\n";
            cout << "-----------------------------------------\n";
            break;
        }
    }
    return 0;
}
```

main()

QuickSort

```
(base) macad@nganhhtuann-3 output % ./"sortAlgo"
-----------------------------------------
(1) QuickSort
(2) MergeSort
(3) NaturalMergeSort
(4) HeapSort
(5) RadixSort
-----------------------------------------
Chon thuat toan muon dung de sap xep mang: 1
-----------------------------------------
Thuat toan Quick Sort:
Mang truoc khi sap xep: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
Thu tu mang moi lan swap vi tri pivot:
Pivot: 6
1 5 2 3 5 1 3 5 3 2 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 5
1 2 2 3 3 1 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 3
1 2 2 3 1 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 2
1 1 2 3 2 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 1
1 1 2 3 2 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 3
1 1 2 2 3 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 3
1 1 2 2 3 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 5
1 1 2 2 3 3 3 5 5 5 34 6 7 345 123 25 42
-----------------------------------------
Pivot: 345
1 1 2 2 3 3 3 5 5 5 34 6 7 42 123 25 345
-----------------------------------------
Pivot: 7
1 1 2 2 3 3 3 5 5 5 7 6 34 42 123 25 345
-----------------------------------------
Pivot: 7
1 1 2 2 3 3 3 5 5 5 6 7 34 42 123 25 345
-----------------------------------------
Pivot: 42
1 1 2 2 3 3 3 5 5 5 6 7 34 25 123 42 345
-----------------------------------------
Pivot: 34
1 1 2 2 3 3 3 5 5 5 6 7 25 34 123 42 345
-----------------------------------------
Pivot: 7
1 1 2 2 3 3 3 5 5 5 6 7 25 34 123 42 345
-----------------------------------------
Pivot: 34
1 1 2 2 3 3 3 5 5 5 6 7 25 34 123 42 345
-----------------------------------------
Pivot: 123
1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Mang sau khi sap xep: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
Time complex: 17 ms
-----------------------------------------
```

MergeSort

```
(base) macad@nganhhtuann-3 output % ./"sortAlgo"
-------------------------------------------
(1) QuickSort
(2) MergeSort
(3) NaturalMergeSort
(4) HeapSort
(5) RadixSort
-------------------------------------------
Chon thuat toan muon dung de sap xep mang: 2
-------------------------------------------
Thuat toan Merge Sort:
Mang truoc khi sap xep: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
Thu tu mang moi lan merge:
Left arr: 1
Right arr: 5
Array: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 2
Right arr: 3
Array: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 25
Right arr: 1
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 3
Right arr: 5
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 6
Right arr: 34
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 2
Right arr: 3
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 7
Right arr: 345
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 123 5 42
-------------------------------------------
Left arr: 123
Right arr: 5
Array: 1 5 2 3 1 25 3 5 6 34 2 3 7 345 5 123 42
-------------------------------------------
Left arr: 1 5
Right arr: 2 3
Array: 1 2 3 5 1 25 3 5 6 34 2 3 7 345 5 123 42
-------------------------------------------
Left arr: 1 25
Right arr: 3 5
Array: 1 2 3 5 1 3 5 25 6 34 2 3 7 345 5 123 42
-------------------------------------------
Left arr: 6 34
Right arr: 2 3
Array: 1 2 3 5 1 3 5 25 2 3 6 34 7 345 5 123 42
-------------------------------------------
Left arr: 7 345
Right arr: 5 123
Array: 1 2 3 5 1 3 5 25 2 3 6 34 5 7 123 345 42
-------------------------------------------
Left arr: 1 2 3 5
Right arr: 1 3 5 25
Array: 1 1 2 3 3 5 5 25 2 3 6 34 5 7 123 345 42
-------------------------------------------
Left arr: 2 3 6 34
Right arr: 5 7 123 345
Array: 1 1 2 3 3 5 5 25 2 3 5 6 7 34 123 345 42
-------------------------------------------
Left arr: 1 1 2 3 3 5 5 25
Right arr: 2 3 5 6 7 34 123 345
Array: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 123 345 42
-------------------------------------------
Left arr: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 123 345
Right arr: 42
Array: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-------------------------------------------
Mang sau khi sap xep: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
Time complex: 25 ms
-------------------------------------------
```

- Natural Merge Sort:

```
(base) macad@nganhhtuann-3 output % ./"sortAlgo"
-----------------------------------------
(1) QuickSort
(2) MergeSort
(3) NaturalMergeSort
(4) HeapSort
(5) RadixSort
-----------------------------------------
Chon thuat toan muon dung de sap xep mang: 3
-----------------------------------------
Thuat toan Natural Merge Sort:
Mang truoc khi sap xep: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
Thu tu mang moi lan merge:
Left arr: 1 5
Right arr: 2 3 25
Array: 1 2 3 5 25 1 3 5 6 34 2 3 7 345 123 5 42
-----------------------------------------
Left arr: 1 3 5 6 34
Right arr: 2 3 7 345
Array: 1 2 3 5 25 1 2 3 3 5 6 7 34 345 123 5 42
-----------------------------------------
Left arr: 123
Right arr: 5 42
Array: 1 2 3 5 25 1 2 3 3 5 6 7 34 345 5 42 123
-----------------------------------------
Left arr: 1 2 3 5 25
Right arr: 1 2 3 3 5 6 7 34 345
Array: 1 1 2 2 3 3 3 5 5 6 7 25 34 345 5 42 123
-----------------------------------------
Left arr: 1 1 2 2 3 3 3 5 5 6 7 25 34 345
Right arr: 5 42 123
Array: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Mang sau khi sap xep: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
Time complex: 22 ms
-----------------------------------------
```

HeapSort

```
(base) macad@nganhhtuann-3 output % ./"sortAlgo"
-----------------------------------------
(1) QuickSort
(2) MergeSort
(3) NaturalMergeSort
(4) HeapSort
(5) RadixSort
-----------------------------------------
Chon thuat toan muon dung de sap xep mang: 4
-----------------------------------------
Thuat toan Heap Sort:
Mang truoc khi sap xep: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
Mang sau khi tao max heap: 345 42 123 6 34 7 3 5 5 25 2 3 1 1 2 3 5
Thu tu mang sau khi swap lan luot voi max heap:
Max heap = 123
123 42 7 6 34 5 3 5 5 25 2 3 1 1 2 3 345
-----------------------------------------
Max heap = 42
42 34 7 6 25 5 3 5 5 3 2 3 1 1 2 123 345
-----------------------------------------
Max heap = 34
34 25 7 6 3 5 3 5 5 2 2 3 1 1 42 123 345
-----------------------------------------
Max heap = 25
25 6 7 5 3 5 3 1 5 2 2 3 1 34 42 123 345
-----------------------------------------
Max heap = 7
7 6 5 5 3 3 3 1 5 2 2 1 25 34 42 123 345
-----------------------------------------
Max heap = 6
6 5 5 5 3 3 3 1 1 2 2 7 25 34 42 123 345
-----------------------------------------
Max heap = 5
5 5 5 2 3 3 3 1 1 2 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 5
5 3 5 2 2 3 3 1 1 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 5
5 3 3 2 2 1 3 1 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 3
3 2 3 1 2 1 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 3
3 2 3 1 2 1 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 3
3 2 1 1 2 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 2
2 2 1 1 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 2
2 1 1 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 1
1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 1
1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Max heap = 1
1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Mang sau khi sap xep: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
Time complex: 21 ms
-----------------------------------------
```

- Radix sort:

```
(base) macad@nganhhtuann-3 output % ./"sortAlgo"
-----------------------------------------
(1) QuickSort
(2) MergeSort
(3) NaturalMergeSort
(4) HeapSort
(5) RadixSort
-----------------------------------------
Chon thuat toan muon dung de sap xep mang: 5
-----------------------------------------
Thuat toan Radix Sort:
Mang truoc khi sap xep: 1 5 2 3 25 1 3 5 6 34 2 3 7 345 123 5 42
Gia tri lon nhat trong mang = 345
Mang sau moi lan thuc hien dem:
Mang dem: 0 0 2 5 9 10 15 16 17 17
1 1 2 2 42 3 3 3 123 34 5 25 5 345 5 6 7
-----------------------------------------
Mang dem: 0 12 12 14 15 17 17 17 17 17
1 1 2 2 3 3 3 5 5 5 6 7 123 25 34 42 345
-----------------------------------------
Mang dem: 0 15 16 16 17 17 17 17 17 17
1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
-----------------------------------------
Mang sau khi sap xep: 1 1 2 2 3 3 3 5 5 5 6 7 25 34 42 123 345
Time complex: 21 ms
-----------------------------------------
```

Nhận xét:

Việc viết chương trình mô phỏng trực quan các thuật toán để tìm hiểu và dễ dàng trogn việc mường tượng mảng sau mỗi step sẽ trở thành như thế nào. Ngoài ra yêu cầu xuất ra độ phúc tạp thời gian cũng rất hữu ích để so sanh hiệu năng của các thuật toán với nhau với input đầu vào khác nhau

Trong đó độ phức tạp thuật toán sẽ được nhận xét như sau:

- QuickSort: trung bình O(nlogn), trường hợp xấu nhất O(n^2)
- MergeSort: trung bình O(nlogn) nhưng yêu cầu bộ nhớ hơn để chứa các mảng phụ, hiệu suất tốt trong mọi trường hợp
- Natural Merge Sort: trung bình O(nlogn) như merge sort giảm thiểu độ phức tạp ở các run
- HeapSort: trung bình O(nlogn) sử dụng tốt trong mọi trường hợp

- RadixSort: O(num(n+k)) trong đó num là số chữ số của số lớn nhất trong mảng, n là số lượng phần tử và k là số lượng giá trị khác nhau mà mỗi chữ số có thể có, thuật toán hiệu quả nhất khi các phần tử trong danh sách có cùng độ dài num.