Lab 2A

Search Algorithms

In this lab session, we will experiment with number of search algorithms.

1 Exercise 1

Write a program to determine **the first found position** of an element in the one-dimensional array of integers with different search algorithms:

1. Linear search (implement with while).
2. Linear search with sentinel (implement with while).
3. Binary search (the array is sorted in ascending order already).
4. Interpolation search (the array is sorted in ascending order already).

Using command line: `Exercise_1.exe algorithm x input.txt output.txt`
For example: `Exercise_1.exe 1 3 input_1.txt output_1.txt`
where:

- `1` is Linear search algorithm.
- `3` is search key.
- `input_1.txt` is input file.
- `output_1.txt` is output file.

Input format:

- The first line contains a single integer N, which is the size of array.
- The next line denotes the array's elements. Each element is separated by a space.

For example:

```
6
2 8 3 9 1 3
```

Output format:
The first found position of x in array (position count from 0). If not found, output contains `-1`.
For example: `2`

2 Exercise 2

Build `WORD` structure to store word in language with name and its definition. Apply any search algorithm to create dictionary software.

Using command line: `Exercise_2.exe word1 word2 word3 ... output.txt`

- `word1`, `word2` and so on are words which user want to know definition.
- `output.txt` is output file which contains their definitions. Each word and its definition is displayed similar with dictionary file.

In the dictionary, each line will store one word and its definitions. Word and definition are separated by the colon. The following figure is an example of dictionary.

```
abalone: bao ngu
abalones: bao ngu
abandon:bom  tu bo, bo roi, ruong bo
abandoned:bi bo roi, bi ruong bo
abandoner: nguoi rut don
abandoners: nguoi rut don
abandoning:bom  tu bo, bo roi, ruong bo
abandonment:su bo, su tu bo, su bo roi, su ruong bo
abandonments:su bo, su tu bo, su bo roi, su ruong bo
abandons:bom  tu bo, bo roi, ruong bo
abapikal: xa dinh, xa ngon
abarticular: hoc ngoai khop, trat khop
abas:ao aba
abase:lam ha pham gia, lam mat the dien, lam nhuc
abased:lam ha pham gia, lam mat the dien, lam nhuc
abasement:su lam ha pham gia, su lam mat the dien, su lam nhuc
abasements:su lam ha pham gia, su lam mat the dien, su lam nhuc
```

Figure 1: Dictionary file.

Lab 2B

Sort Algorithms (Part 1)

In this lab session, we will experiment with some sort algorithms.

3 Exercise 1

Implement sort algorithms include **insertion sort**, **selection sort** and **interchange sort** for integer array with random values.
Compare the system time on a random array of size 10000 or more. Note that all algorithms should use the same data to ensure fairness when comparing.

Example output:

```
Insertion Sort Time: 0.062 seconds
Selection Sort Time: 0.094 seconds
Interchange Sort Time: 0.344 seconds
The fastest algorithm: Insertion Sort
The slowest algorithm: Interchange Sort
```

4 Exercise 2

Sort an integer array with even numbers in ascending order and odd numbers in descending order. Note that, positions of even and odd parts is no change. For example:

Input:
`9`
`1 2 4 7 15 18 8 9 11`
Output:
`15 2 4 11 9 8 18 7 1`

Only **one** algorithm is required to be implemented for this exercise. Please choose the most optimal algorithm to implement.

5 Exercise 3

Read the data from the `employee.txt` and sort on age in descending order using **insertion sort**, **selection sort** and **interchange sort**. If age is equal, using names in alphabetical order.
Write the sorted data to another file `sortedemponage.txt`.

Please note that:

- Implement all 3 algorithms, and create a menu to select them when executing the program.
- If the birthday years are too big, please correct them according to the last two numbers.

  For example: `120433` to `1933`.
- Use `time.h` or `ctime` library to get current year.

6 Exercise 4

Sort an arbitrary two-dimensional array in spiral order as following image:

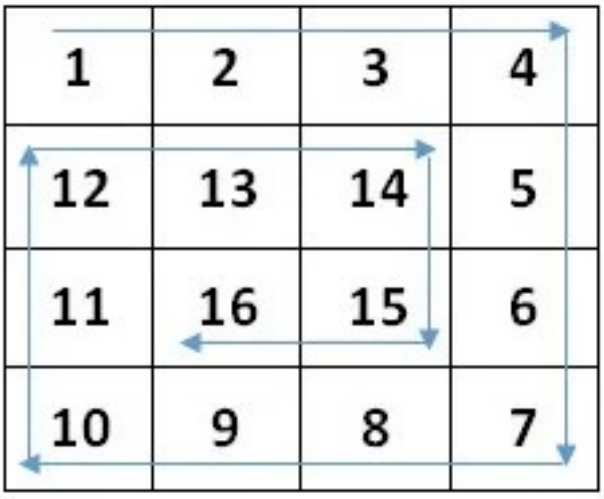

Figure 2: Spiral order.

Please note that the two-dimensional array does not have to be square matrix.

Lab 2C

Sort Algorithms (Part 2)

In this lab session, we will continue to experiment the sorting algorithms you have learned.

7 Exercise 1

Implement sort algorithms for integer array with random values, include:

- Heap sort.
- Quick sort.

Compare the system time on a random array of size 100000 or more. Note that all algorithms should use the same data to ensure fairness when comparing.

Example output:

```
Heap Sort: 0.046 seconds
Quick Sort: 0.016 seconds
The fastest algorithm: Quick Sort
The slowest algorithm: Heap Sort
```

8 Exercise 2

Continue Exercise 1 with these following sort algorithms:

- Merge sort.
- Counting Sort.
- Radix Sort.
- Flash Sort.

Compare the system time on a random array of size 100000 or more. Note that all algorithms should use the same data to ensure fairness when comparing.

9 Exercise 3

Performs the following operations encountered when processing data for an admission exam:

1. Calculate the average scores for the candidates.
2. Assign places to the accepted candidates, based on `m` options they have, and print the list.
3. List all rejected candidates in descending order.

The exam is composed of two tests, graded with reals in the range [1, 10]. When average scores (truncated to two decimal positions right of the point) are equal, the score of the first test and then the scores of the second test is used to decide position. If equality persists, increase the number of available positions from a certain option.

Input format:

- Number of options `m` alone on one line.
- Pairs option maximum number of admitted candidates, separated by blanks, each pair on one line.
- Candidate data, one candidate on each line.

For example:

```
4
1 25
2 30
3 35
4 20
"Doe John",9.30,9.80,4,2,1,3
"Doe Jane",9.70,9.70,1,2,4,3
...
```

Output format:

- Successful candidates for each option.
- Unsuccessful candidates.

For example:

```
Successful candidates for option 1
1. Doe Jane 9.70
...
...
Successful candidates for option 4
1. Doe John 9.55
Unsuccessful candidates
1. Jones Jim 4.99
...
```