Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

Алгоритми та структури даних

Лабораторна робота №2

Виконав: студент групи ФБ-82
Козачок Вячеслав
Перевірила:
Лавренюк

Київ - 2020

Мета: Познайомитися з роботою поширених методів сортування, з критеріями та методикою їх порівняння.

Варіант: $9 - 6 = 3$

- Сортування методом бульбашки;
- Сортування методом Шелла.

Хід роботи:

Не відсортований масив:

```
9384 887 2778 6916 7794 8336 53 9384 887 2778 6916 7794 8336 5387 493 6650
1422 2363 28 869160 7764 3927 541 3427 9173 5737 5212 5369 2568 6430 5783
1531 2863 5124 4068 3136 3930 9803 4023 3059 3070 8168 1394 8457 5012 8043
6230 7374 4422 4920 3785 8538 5199 4325 8316 4371 6414 3527 6092 8981 9957
1874 6863 9171 6997 7282 2306 926 7085 6328 337 6506 847 1730 1314 5858 6125
3896 9583 546 8815 3368 5435 365 4044 3751 1088 6809 7277 7179 5789 3585
5404 2652 2755 2400 9933 5061 9677 3369 7740 13 6227 8587 8095 7540
```

Відсортований масив:

```
13 28 60 337 365 493 541 546 847 887 926 1088 1314 1394 1422 1531 1730 1874
2306 2363 2400 2568 2652 2755 2778 2863 3059 3070 3136 3368 3369 3427 3527
3585 3751 3785 3896 3927 3930 4023 4044 4068 4325 4371 4422 4920 5012 5061
5124 5199 5212 5369 5387 5404 5435 5737 5783 5789 5858 6092 6125 6227 6230
6328 6414 6430 6506 6650 6809 6863 6916 6997 7085 7179 7277 7282 7374 7540
7740 7764 7794 8043 8095 8168 8316 8336 8457 8538 8587 8691 8815 8981 9171
9173 9384 9583 9677 9803 9933 9957
```

Результати порівняння методів сортування

	Bubble Sort					**Shell Sort**				
N	**К-ть копіювань (M)**		**К-ть порівнянь (C)**		**Час (T)**	**К-ть копіювань (M)**		**К-ть порівнянь (C)**		**Час (T)**
	Теорет.	**Експерим.**	**Теорет.**	**Експерим.**		**Теорет.**	**Експерим.**	**Теорет.**	**Експерим.**	
100	2475	7059	4950	7503	$6.3*10^{-5}$	251	783	251	1188	$2.0*10^{-5}$
1000	249750	749493	499500	751331	$330*10^{-5}$	3981	16000	3981	23039	$35.6*10^{-5}$
10000	24997000	55171710	49995000	68405570	0.280	63095	359264	63095	354149	0.0053

Code

```cpp
#include <iostream>
#include <fstream>
#include <ctime>
#include <cmath>
#include <random>

void printArray(int * a, int n);

void insertionSort(int * arr, int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;

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

void bubbleSort(int * arr, int n)
{
    int compare = 0, move = 0;
    for(int i = 0; i < n; i++)
    {
        compare++;
        for(int k = i; k < n; k++)
        {
            compare++;
            if(arr[i] > arr[k])
            {
                compare++;
                int num = arr[i];
                arr[i] = arr[k];
                arr[k] = num;
                move += 3;
            }
        }
    }
    std::cout << "Compares: " << compare << ", Moves: " << move << std::endl;

}


void shellSort(int * arr, int n)
{


    int stepsNum = (int)log2(n) - 1 ;
    int * steps = new int[stepsNum];
    steps[stepsNum-1] = 1;

    for(int i = stepsNum-2;i >= 0; i--)
    steps[i] = 2 * steps[i+1] + 1;

    std::cout << stepsNum << "; ";
    printArray(steps, stepsNum);


    int i, j, increment, temp;
    int move = 0, compare = 0;

    // for(increment = n/2; increment > 0; increment/=2)
    for(int k = 0; k < stepsNum; k++ )
    {
        compare++;
```

```
72          move++;
73          increment = steps[k];
74          for(i = increment; i < n; i++)
75          {
76              compare++;
77              temp = arr[i];
78              for(j = i; j >= increment ;j-=increment)
79              {
80                  //perform the insertion sort for this section
81                  compare++;
82                  if(temp < arr[j-increment])
83                  {
84                      move++;
85                      arr[j] = arr[j-increment];
86                  }
87                  else
88                  {
89                      break;
90                  }
91              }
92              move++;
93              arr[j] = temp;
94          }
95      }
96      std::cout << "Compares: " << compare << ", Moves: " << move << std::endl;
97
98  }
99
100 void printArray(int * a, int n)
101 {
102     for(int i = 0; i < n; i++)
103     {
104         std::cout << a[i] << ' ';
105     }
106     std::cout << std::endl;
107 }
108
109 int *  fillArray(int n)
110 {
111     int * arr = new int[n];
112     for(int i = 0; i < n; i++)
113     {
114         arr[i] = rand() % 10000 + 1;
115     }
116     return arr;
117 }
118
119
120 void call(void (&func)(int* arr , int n), int * arr, int n)
121 {
122     clock_t t = clock();
123     (*func)(arr, n);
124     std::cout << "Sort time: " << ((double)(clock() - t))/CLOCKS_PER_SEC << std::endl;
125 }
126
127 int main(int argv, char** argc) {
128
129     int n = 100;
130     int * arr = fillArray(n);
131     std::cout << "1. Bubble Sort\n2. Shell Sort" << std::endl;
132     int choice;
133
134     std::cin >> choice;
135
136     printArray(arr, n);
137     if(choice == 1) call(bubbleSort, arr, n);
138     if(choice == 2) call(shellSort, arr, n);
139     printArray(arr, n);
140
141     return 0;
142 }
```

Висновки

Під часвиконання роботи, я вивчив такі алгоритми сортування як `Shell Sort`, `Insertion Sort` та `Bubble Sort`. Навчився заміряти час роботи алгоритмів, робити теоретичну оцінку їх швидкодії. Навчився передавати функцію, як аргумент у іншу функцію.