Homework 3

Hamza Bouhelal

February 2021

Problem 3.1

a) f(n) = 9n and g(n) = 5n^3
First, let's compute $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{9n}{5n^3} = \lim_{n\to\infty} \frac{9}{5n^2} = 0$
Then, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{5n^3}{9n} = \lim_{n\to\infty} \frac{5n^2}{9} = +\infty$

We conclude that:
$f \in o(g)$, $f \in O(g)$, $f \notin \Omega(g)$, $f \notin \Theta(g)$, $f \notin w(g)$,
And that:
$g \notin o(f)$, $g \notin O(f)$, $g \in \Omega(f)$, $g \notin \Theta(f)$, $g \in w(f)$.

b) f(n) = 9n$^{0.8}$ $+ 2n^{0.3} + 14\ log(n)$ and $g(n) = \sqrt{n}$,
First, let's compute $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{9n^{0.8}+2n^{0.3}+14\times log(n)}{\sqrt{n}} = +\infty$
Then, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{\sqrt{n}}{9n^{0.8}+2n^{0.3}+14\times log(n)} = 0$

We conclude that:
$f \notin o(g)$, $f \notin O(g)$, $f \in \Omega(g)$, $f \notin \Theta(g)$, $f \in w(g)$,
And that:
$g \in o(f)$, $g \in O(f)$, $g \notin \Omega(f)$, $g \notin \Theta(f)$, $g \notin w(f)$.

$c)f(n)$ =n$^2/\log n$ and $g(n) = n \times log(n)$,
First, let's compute $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\frac{n^2}{\log n}}{n\times log(n)} = \lim_{n\to\infty} \frac{n}{log(n)^2} = +\infty$
Then, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{n\times log(n)}{\frac{n^2}{\log n}} = \lim_{n\to\infty} \frac{log(n)^2}{n} = 0$

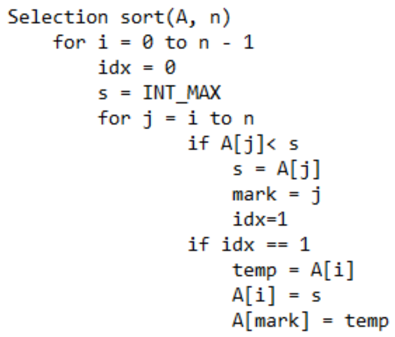
We conclude that:
$f \notin o(g)$, $f \notin O(g)$, $f \in \Omega(g)$, $f \notin \Theta(g)$, $f \in w(g)$,
And that:
$g \in o(f)$, $g \in O(f)$, $g \notin \Omega(f)$, $g \notin \Theta(f)$, $g \notin w(f)$.

$d) f(n)$ =log(3n)^3 and $g(n) = 9 \times log(n)$,
First, let's compute $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{log(3n)^3}{9\times log(n)} = +\infty$
Then, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{9\times log(n)}{log(3n)^3} = 0$

We conclude that:
$f \notin o(g)$, $f \notin O(g)$, $f \in \Omega(g)$, $f \notin \Theta(g)$, $f \in w(g)$,
And that:
$g \in o(f)$, $g \in O(f)$, $g \notin \Omega(f)$, $g \notin \Theta(f)$, $g \notin w(f)$.

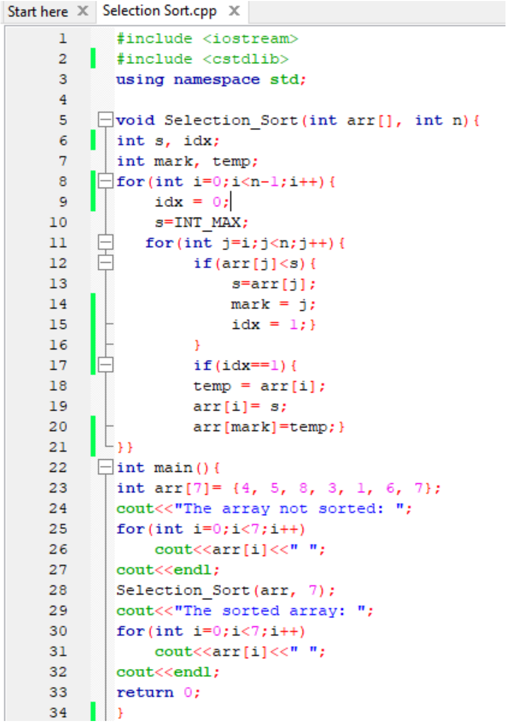
Problem 3.2

a)



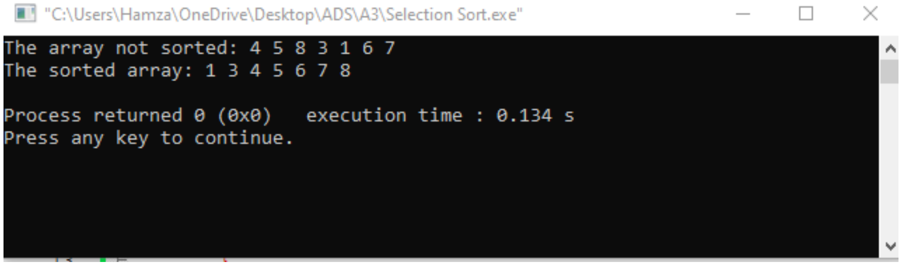
```
Selection sort(A, n)
    for i = 0 to n - 1
        idx = 0
        s = INT_MAX
        for j = i to n
                if A[j]< s
                    s = A[j]
                    mark = j
                    idx=1
                if idx == 1
                    temp = A[i]
                    A[i] = s
                    A[mark] = temp
```

b)



```
#include <iostream>
#include <cstdlib>
using namespace std;

void Selection_Sort(int arr[], int n){
int s, idx;
int mark, temp;
for(int i=0;i<n-1;i++){
     idx = 0;
     s=INT_MAX;
   for(int j=i;j<n;j++){
          if(arr[j]<s){
              s=arr[j];
              mark = j;
              idx = 1;}
          }
          if(idx==1){
          temp = arr[i];
          arr[i]= s;
          arr[mark]=temp;}
}}
int main(){
int arr[7]= {4, 5, 8, 3, 1, 6, 7};
cout<<"The array not sorted: ";
for(int i=0;i<7;i++)
     cout<<arr[i]<<" ";
cout<<endl;
Selection_Sort(arr, 7);
cout<<"The sorted array: ";
for(int i=0;i<7;i++)
     cout<<arr[i]<<" ";
cout<<endl;
return 0;
}
```

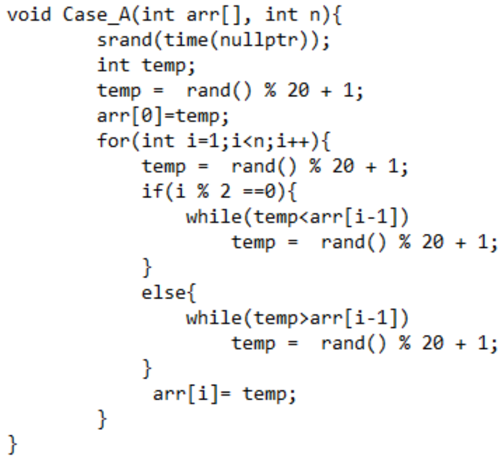


```
The array not sorted: 4 5 8 3 1 6 7
The sorted array: 1 3 4 5 6 7 8

Process returned 0 (0x0)   execution time : 0.134 s
Press any key to continue.
```

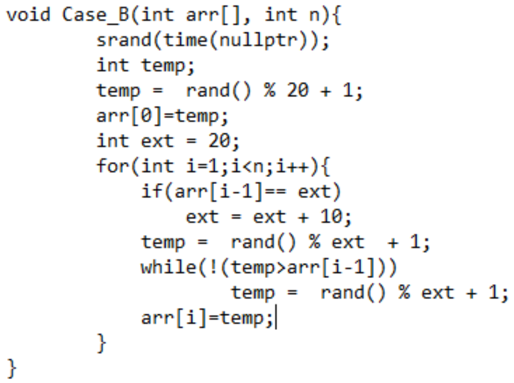
c)

In order to generate a random input sequences of length n that involves tho most swaps (Case A), we are looking for n-1 swaps at the Selection Sort, the only way of getting such a result would be sorting an array where each element is either smaller or bigger than the element before, depending on the position that element is in.



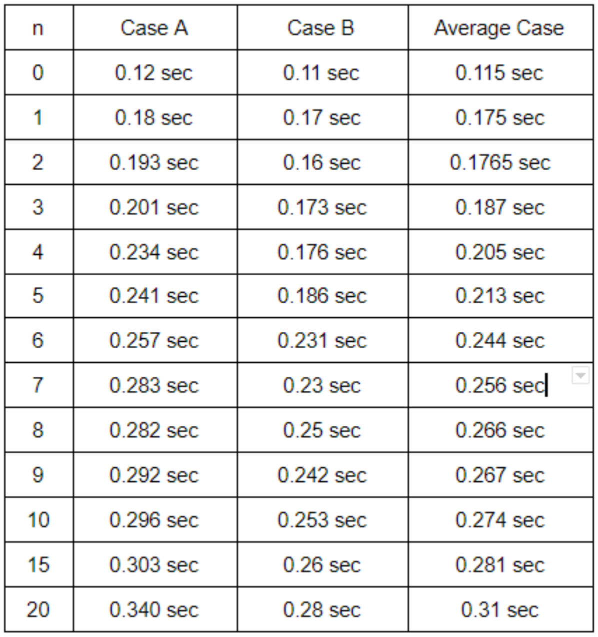
```
void Case_A(int arr[], int n){
        srand(time(nullptr));
        int temp;
        temp =  rand() % 20 + 1;
        arr[0]=temp;
        for(int i=1;i<n;i++){
            temp =  rand() % 20 + 1;
            if(i % 2 ==0){
                while(temp<arr[i-1])
                    temp =  rand() % 20 + 1;
            }
            else{
                while(temp>arr[i-1])
                    temp =  rand() % 20 + 1;
            }
             arr[i]= temp;
        }
}
```

In order to generate a random input sequences of length n that involves tho least swaps (Case B), we are looking here for 0 swaps, to get such a result each element must be smaller than the next element, in other words, the array must already be sorted.



```
void Case_B(int arr[], int n){
        srand(time(nullptr));
        int temp;
        temp =  rand() % 20 + 1;
        arr[0]=temp;
        int ext = 20;
        for(int i=1;i<n;i++){
            if(arr[i-1]== ext)
                ext = ext + 10;
            temp =  rand() % ext  + 1;
            while(!(temp>arr[i-1]))
                    temp =  rand() % ext + 1;
            arr[i]=temp;
        }
}
```

d)



n	Case A	Case B	Average Case
0	0.12 sec	0.11 sec	0.115 sec
1	0.18 sec	0.17 sec	0.175 sec
2	0.193 sec	0.16 sec	0.1765 sec
3	0.201 sec	0.173 sec	0.187 sec
4	0.234 sec	0.176 sec	0.205 sec
5	0.241 sec	0.186 sec	0.213 sec
6	0.257 sec	0.231 sec	0.244 sec
7	0.283 sec	0.23 sec	0.256 sec
8	0.282 sec	0.25 sec	0.266 sec
9	0.292 sec	0.242 sec	0.267 sec
10	0.296 sec	0.253 sec	0.274 sec
15	0.303 sec	0.26 sec	0.281 sec
20	0.340 sec	0.28 sec	0.31 sec