## Laborator 8

Funcţii utilizate:

1. **tic** - porneşte un cronometru pentru a monotoriza performanţa execuţiei.
2. **toc** - afişează câte secunde s-au scurs după execuţia comenzii tic

Exemplu:

```
>> tic
>> toc % am introdus comanda dupa cateva secunde
Elapsed time is 2.704124 seconds.
```

**Problema I**

**Construim algoritmul de QuickSort:**

```
function  A=QuickSort(A,p,r)
i=p;
j=p;
 %  A(r) reprezinta pivotul
while j<=r
    if A(j)>A(r) % element mai mare decat pivotul, crestem j
        j=j+1;
    elseif A(j)<=A(r) % element mai mic decat pivotul
                      % mutam acest element in primul subvector,
                      % care va deveni A(p:q-1)
        inter=A(j);
        A(j)=A(i);
        A(i)=inter;
        i=i+1;
        j=j+1;

    end
end
q=i-1; %am identificat pozitia finala a pivotului in vectorul sortat

if p<r

    A1=QuickSort(A,p,q-1); %aplicam algoritmul pt suvectorul A(p:q-1)
```

```
    A(p:(q-1))=A1(p:(q-1));

    A2=QuickSort(A,q+1,r); %aplicam algoritmul pt suvectorul A(q+1:r)
    A(q+1:r)=A2(q+1:r);
end


end
```

**Exemplu de apel:**

```
>> [A,n]=QuickSort([2 8 7 1 3 5 6 2], 1, 8)

A =

     1     2     2     3     5     6     7     8


n =

   3.3639e-05
```

**Problema II**

**Generam aleator un indice cuprins între valorile $a$ şi $b$:**

```
>>t_real=unifrnd(a,b); %obtinem un nr real aleator intre a si b
>>t_intreg=round(t_real)
```

**Următorul pas este generarea unui indice între $p$ şi $r$, urmată apoi de interschimbarea elementului de pe acea poziţie aleatoare cu elementul de pe poziţia $r$:**

```
indice=round(unifrnd(p,r));
```

```
aux=A(indice);
A(indice)=A(r);
A(r)=aux;
```

**Vom simula sortarea a $N$ vectori de lungime data $l$:**

```
clear all

%simulare sortare vectori de lungime l
l=10;
N=100;
timp=zeros(1,N);
timp1=zeros(1,N);
for i=1:N
    A=randperm(l);%generam o permutare aleatoare de ordin l
    tic;
    A1=QuickSort(A,1,length(A));
    timp(i)=toc;

    tic;
    A2=RandomizedQuickSort(A,1,length(A));
    timp1(i)=toc;
end
%valoarea medie a timpului de executie pentru cele doua metode de sortare

fprintf('%.8f\n',mean(timp));

fprintf('%.8f\n',mean(timp1));
```

**Problema III**

**Construim algoritmul de QuickSelect:**

```
function  A=QuickSelect(A,p,r,indice_cautat)
```

```
i=p;
j=p;
while j<=r
    if A(j)>A(r) %element mai mic decat pivotul

        j=j+1;

    elseif A(j)<=A(r) % element mai mare decat pivotul
         inter=A(j);
        A(j)=A(i);
        A(i)=inter;
        i=i+1;
        j=j+1;

    end
end
q=i-1;

if q==indice_cautat %al i-lea cel mai mic element se afla pe pozitia pivotului


    return;

end
if q>indice_cautat && p<r

    A1=QuickSelect(A,p,q-1,indice_cautat);
    A(p:(q-1))=A1(p:(q-1));

elseif q<indice_cautat && p<r

    A2=QuickSelect(A,q+1,r,indice_cautat);
    A(q+1:r)=A2(q+1:r);

end

end
```