Seminar 1 – Kompleksiteti

Pyetje Teorike

1. Cilat janë dy mënyrat për analizën e një algoritmi?
2. Për çfarë shërben metoda e Java currentTimeMillis()?
3. Cilat cilësohen si veprime elementare në një algoritëm?
4. Çfarë është Big-O notation?
5. Cilët janë disa faktorë që ndikojnë në kohën e ekzekutimit të një algoritmi?

Ushtrime

1. Shkruani algoritmin për kërkimin linear të një elementi x në një vektor a[n] . Shprehni funksionin T(n) të numrit të instruksioneve për rastin më të keq. Gjeni kompleksitetin për rastin më të mirë dhe atë më të keq.

```
int kërko(int a[], int n, int x) {
    int i;
    int ugjet = 0;
    for(i=0; i<n; i ++) {
       if(a[i]==x) {
          ugjet = 1;
          break;
       }
    }
    return ugjet;
}
```

2. Jepet vektori A me numra të plotë (pozitiv ose negativ) me n elemente. Ndërtoni algoritmin që gjen sekuencën e elementëve të vektorit që ka shumën maksimale. Nqs të gjithë vlerat e vektorit janë numra negativ, shuma maksimale do të na dalë 0. Gjeni kompleksitetin e algoritmit.

```
int maxSubSum1(const vector<int>& A) {
    int shumaMax =0;
    for (int i=0; i<A.size(); i++) {
       for (int j=i; j <A.size(); j++) {
          int kjoShume = 0;
          for (int k=i; k<=j; k++)
             kjoShume += A[k];
          if (KjoShume>ShumaMax)
             shumaMax = kjoShume;
       }
    }
    return shumaMax;
 }
```

Një mënyre e dytë për të zgjidhur problemin e mësipërm jepet nëpërmjet kodit:
Sa do të jetë kompleksiteti në këtë rast?

```
int maxSubSum2( int A[], int n )
  {
    int shumaMax = 0;
    for (int i=0; i< n; i++) {
      int kjoShume =0;
      for (int j=i; j< n j++) {
        kjoShume += A[j];
        if (kjoShume>shumaMax)
          shumaMax = kjoShume;
      }
    }
    return shumaMax;
  }
```

3. Llogarisni kompleksitetin e algoritmeve të mëposhtëm kur dihet që m dhe n janë integers.

a. Algoritmi A	b. Algoritmi B	c. Algoritmi C

i:=1; j:=1;
While (i<m) or (j<n) do
i:=i+1 ; j:=j+1
end While
	i:=1; j:=1;
	While (j<n) do
	If (i<m) then i:=i+1
	Else j:=j+1
	end If
	End While
		i:=1; j:=1;
		While (j<=n) do
		If (i<=m) then i:=i+1
		else j:=j+1; i:=1
		end If
		end While

4. Tregoni cili është kompleksiteti i veprimeve të shtimit, shumëzimit dhe transpozimit në një matricë nxn?

Rasti a.

```
for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
        a[i][j] = b[i][j] + c[i][j];
```

Rasti b.

```
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    for (k = a[i][j] = 0; k < n; k++)
      a[i][j] += b[i][k] * c[k][j];
```

Rasti c.

```
for (i = 0; i < n - 1; i++)
    for (j = i+1; j < n; j++) {
        tmp = a[i][j];
        a[i][j] = a[j][i];
        a[j][i] = tmp;
    }
```

5. Për secilin nga kodet e mëposhtme bëni një analizë të kohës së ekzekutimit dhe kompleksitetit.

a.
```
sum = 0;
for( i = 0; i < n; ++i )
        ++sum;
```

b.
```
sum = 0;
for( i = 0; i < n; ++i )
        for( j = 0; j < n; ++j )
                ++sum;
```

c.
```
sum = 0;
for( i = 0; i < n; ++i )
        for( j = 0; j < n *n; ++j )
                ++sum;
```

d.
```
sum = 0;
for( i = 0; i < n; ++i )
        for( j = 0; j < i; ++j )
                ++sum;
```

e.
```
sum = 0;
for( i = 0; i < n; ++i ) n
        for( j = 0; j < i * i; ++j ) n^2 *n
                for( k = 0; k < j; ++k ) n^2*n^2*n
                        ++sum;
```

6. Cili është kompleksiteti i funksionit që gjen numrin e k-të më të vogël në një vektor numrash të parenditur?

```
int minK(int a[], int k,int n){
    int i,j,min,tmp;
    for(i=0;i<k;i++){
            min=i;
            for(j=i+1;j<n;j++)
                    if(a[j]<a[min])
                            min=j;
            tmp=a[i];
            a[i]=a[min];
            a[min]=tmp;
    }
    return a[k-1];
}
```

7. Diskutoni mbi kompleksitetin **<u>në kohë</u>** dhe **<u>në hapësirë</u>** të funksionit rekursiv të llogaritjes së faktorialit.

```
int factorial(int n)
{
    if(n >0)
            return n*factorial(n-1);
    elseif ( n==0)
            return 1;
    else
            return -1;
}
```

8. Nëse në një cikël indeksi do të dyfishohet në çdo interacion derisa të arrijë një vlerë x. Sa do të jetë komplesiteti i këtij cikli?

```
int  n=1;
int x;
while(n < =x)
{
   statements;
    n=2*n;
 }
```

9. Nëse numri i interacioneve të ekzekutimit të një cikli do të zvogëlohet në mënyrë konstante në çdo iteracion. Sa do të jetë kompleksiteti i këtij cikli?

```
h=n;
while(h > 0)
{
    for(i=0;i <n;i++)
            statements;
    h=h/2;
}
```

10. Gjeni kompleksitetin në kohë, në varësi të inputit N në kodet e mëposhtme.

Rasti a.

```
int sum=0;
for(int n=N;n>0;n/=2)
        for(int i=0;i<n;i++)
                sum++;
```

Rasti b.

```
int sum=0;
for(int i=1;i<N;i*=2)
        for(int j=0;j<i;j++)
```

```
            sum++;
```

Rasti c.

```
int sum=0;
for(int i=1;i<N;i*=2)
    for(int j=0;j<N;j++)
        sum++;
```

Rasti d.

```
void function(int n)
{
    int count=0;
    for (int i=n/2;i<=n;i++)
    {
        for(int j=1;j<=n;j+2*j)
        {
            for(int k=1;k<=n;k=k*2)
            {
                count++;
            }
        }
    }
}
```

11. Gjeni kompleksitetin në hapësirë.

```
int sum(int arr[], int n)
{
    int sum=0;
    for(int i=0;i<n;i++)
    {
        sum+=arr[i];
    }
    return sum;
}
```

12. Shkruani një program ku jepen dy vektorë të rradhitur me N vlera. Printoni të gjithë vlerat që gjenden në të dy vektorët në mënyrë të rradhitur. Koha e ekzekutimit duhet të jetë O(N) në rastin më të keq.

13. Supozoni se **list** është një objekt i tipit *LinkedList* që përmban një sekuencë prej **n** karakteresh. *(Në Java, tipi i të dhënave LinkedList përfaqëson një sekuencë me elemente duke përdorur një doubly linked list edhe duke patur referenca tek nyja e parë edhe e fundit)*. Për secilin fragment kodi tregoni sa është kompleksiteti në kohë (Big O)? Shpjegoni qartë përgjigjen.

Rasti a.

```
// Knuth shuffle
for (int i = 0; i < list.size(); i++) {
   int r = (int) (Math.random() * (i + 1));
   char c1 = list.get(r); // get element r
   char c2 = list.get(i); // get element i
   list.set(r, c2); // replace element r
   list.set(i, c1); // replace element i
}
```

Rasti b.

```
// palindrome?
boolean isPalindrome = true;
while (list.size() > 1) {
   char c1 = list.removeFirst();
   char c2 = list.removeLast();
   if (c1 != c2) isPalindrome = false;
}
```

Rasti c.

```
// create a reverse copy of the list
LinkedList<Character> copy = new LinkedList<Character>();
for (char c : list) copy.addFirst(c);
```

14. Llogaritni kompleksitetin në **kohë** edhe në **hapësirë** për kodin e mëposhtëm. Shpjegoni qartë përgjigjen.

```
int sum (int n){
   if (n <= 0) {
      return 0;
   }
   return n + sum(n - 1);
}
```