

Opgave 1 (8 %)

Skriv en rekursiv algoritme som kan udskrive summen af de enkelte cifre i et heltal

Eksempel: kaldt med 1024 returneres værdien 7.

Mulig signatur: `int sumOfDigits(int n);`

SVAR

```
private static int number = 1024;
public static void Main(string[] args)
{
    Console.WriteLine(SumOfDigits(number));
}
public static int SumOfDigits(int n)
{
    if (n == 0)
    {
        return 0;
    }

    return (n % 10) + SumOfDigits(n / 10);
}
```

Opgave 2 (12 %)

Betragt følgende algoritme/metode, hvori der indgår tre for-loops:

```
public static int bigOh(int n)
{
    int x = 0;
    for (int i = 1; i <= n; i *= 2) // O(log(n))
    {
        for (int j = 1; j <= n; j++) // O(n)
        {
            for (int k = 1; k <= j; k += i) // O(j/i)
            {
                x++;
            }
        }
    }
    return x;
}
```

Angiv algoritmens tidskompleksitet i Store-O notationen og begrund dit svar for hver enkelt loop.

Hvis du er i tvivl, om dit svar er korrekt, kan det anbefales at teste metoden med forskellige potenser af 2 i sekvens, fx. 2, 4, 8, 16, 32, 64 ... 1024.

SVAR

```
for (int i = 1; i <= n; i *= 2) // O(log(n))
for (int j = 1; j <= n; j++) // O(n)
for (int k = 1; k <= j; k += i) // O(j/i)
```

total $O(n^2)$

```
5
20
76
288
1104
4288
16832
66560
```

Opgave 3 (20 %)

Denne opgave går ud på at tilføje en funktionalitet til den udleverede kode *CriticalPath/KritiskVej*.

Funktionaliteten skal kunne beregne den samlede mængde af slæk (slack) i hele projektet.

Slæk kan defineres som det antal tidsenheder, der er til rådighed til at udføre aktiviteten minus aktivitetens eget tidsforbrug. I det udleverede data tager event 1 6 (seks) tidsenheder, nemlig den længstvarende aktivitets tidsforbrug, som er 6 tidsenheder (se nedenfor). Aktivitet A har 3 tidsenheders slæk – forskellen mellem event 1's længste aktivitet (B), som er 6 og A's tidsforbrug, som er 3. Aktivitet C's slæk er 2 tidsenheder (forskellen mellem 6 og 4).

1; A; 3	slæk 3
1; B; 6	slæk 0
1; C; 4	slæk 2
2; D; 5	slæk 0
3; E; 4	slæk 0
3; F; 1	slæk 3
4; G; 2	slæk 5
4; H; 7	slæk 0
4; I; 1	slæk 6
5; J; 4	slæk 0

Det samlede slæk udskrives, og det korrekte svar i ovenstående eksempel er 19 tidsenheder.

Tip: Løs opgaven i en separat metode med aktivitetstabellen som parameter og genbrug af dele af koden fra main-metoden. Opgaven bliver nemmere, hvis man sorterer aktiviteterne.

Hvis du får problemer med koden, så skriv en detaljeret forklaring på, hvordan opgaven kan løses i pseudokode lignende sprog. En besvarelse uden kode kan højst give 12 ud af 20 points.

```
private static int slackCalculate(List <Aktivitet> table, int eventIndex){

    int totalSlack = 0;
    Aktivitet largestNum = table.getFirst();

    if(table.getLast().getEvent() < eventIndex){
        return 0;
    }

    for (Aktivitet aktivitet : table) {
        if(aktivitet.getEvent() == eventIndex){
            if(aktivitet.getDuration() > largestNum.getDuration())
                largestNum = aktivitet;
        }
    }

    for (Aktivitet aktivitet : table) {
        if(aktivitet.getEvent() == eventIndex){
            totalSlack += largestNum.getDuration() - aktivitet.getDuration();
        }
    }

    return totalSlack + slackCalculate(table, eventIndex + 1);
}
```

Opgave 4 (25 %)

Denne opgave går ud på at flytte rundt på elementerne i et array af heltal og skal have følgende signatur i Java:

```
public static void rotate(int[] arr, int k)
```

Første parameter er arrayet, som skal behandles; den anden parameter angiver det antal elementer, som skal roteres, det vil sige flyttes fra slutningen af arrayet til begyndelsen.

Med parametrene arr: {1,2,3,4,5,6,7} og k: 3, bliver arrayet til {5,6,7,1,2,3,4}, altså de 3 sidste elemester rykkes op, så de står forrest.

For at få maksimumpoints (25) må du kun arbejde på ét array. Alle andre løsninger kan maksimalt give 15 points.

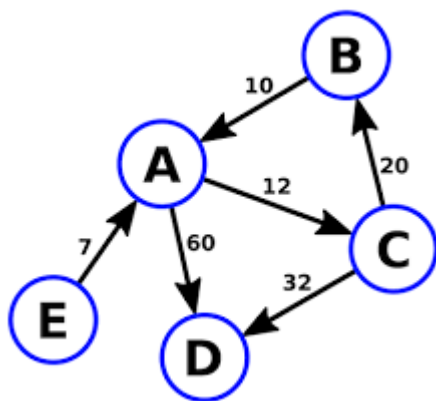
Tip: skriv en metode, som kan vende arrayet eller en del af arrayet om, så tallene står i omvendt rækkefølge.

Besvarelser i C++ må gerne anvende `vector` i stedet for et array.

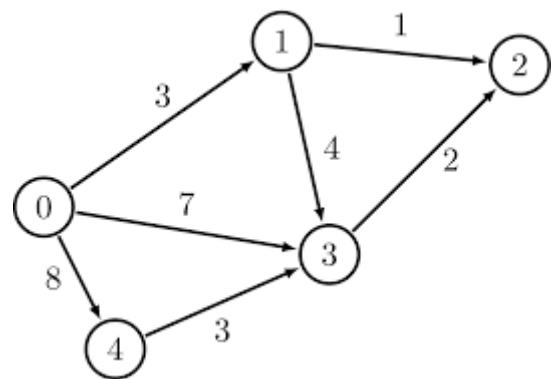
```
public static void rotateFirst(int[] arr, int k) //First index to last k times
{
    for (int i = 0; i < k; i++)
    {
        int first = arr[0];
        for (int j = 0; j < arr.Length - 1; j++)
        {
            arr[j] = arr[j + 1];
        }
        arr[arr.Length - 1] = first;
    }

    Console.WriteLine("Rotate First array: ");
    foreach (int i in arr)
    {
        Console.Write(i);
    }
}
```

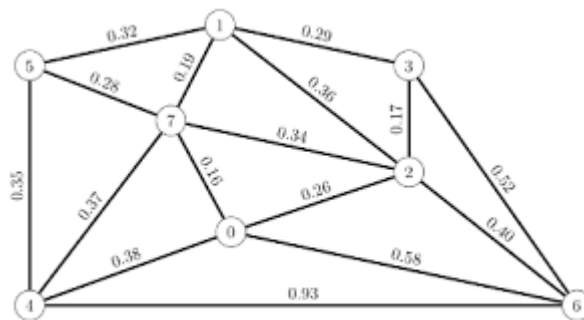
Opgave 5 (20 %)



Graf 1



Graf 2



Graf 3

I denne opgave skal du arbejde med de ovenstående tre grafer.

Delopgave 1 (12 %)

Karakteriser de tre grafer med hensyn til de egenskaber, vi har gennemgået (directed, weighted, dense, connected etc.) og angiv grafernes anvendelighed i forhold til de fire algoritmer, vi har gennemgået:

- Topologisk sortering
- Dijkstra
- Kruskal
- Prim

Graf 1: Directed, Weighted, Cyclic, Sparse, Simple

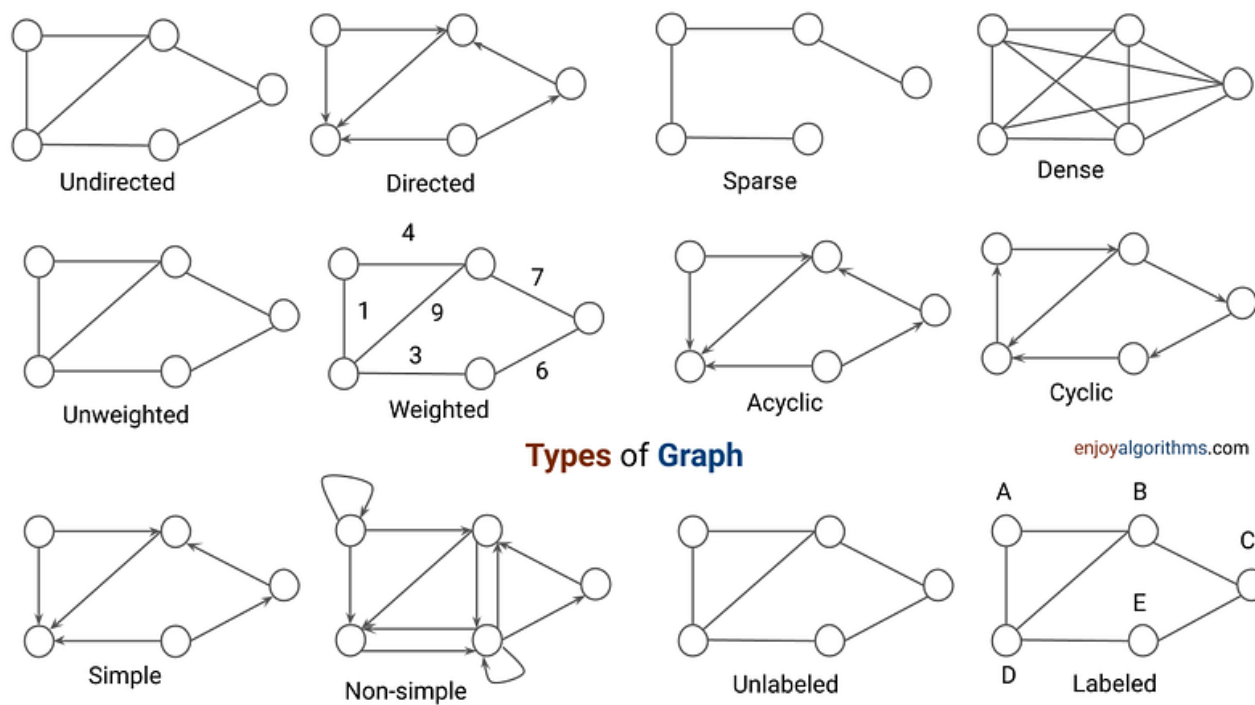
Graf 1: Dijkstra

Graf 2: Weighted, Directed, Acyclic, Simple

Graf 2: Topologisk sortering, Dijkstra

Graf 3: Dense, Weighted, Connected, Undirected,

Graf 3: Kruskal, Prim



Delopgave 2 (8 %)

Anvend Dijkstras algoritme på Graf 1 med start i node E.

Opgave 6 (10 %)

Nedenstående hashtabel har plads til 17 elementer.

De første otte indsættelser hasher alle til indeks 5 og er blevet placeret ved anvendelse af *quadratic probing*.

Ét af dem er placeret forkert. Hvilket element er der tale om, og hvilket indeks er den korrekte placering?

0	
1	
2	
3	G
4	F
5	C
6	L
7	
8	P
9	U
10	
11	
12	
13	Y
14	M
15	
16	

P is wrongly placed. It needs to be on index 7 and not index 8

Opgave 7 (5 %)

Den nedenstående tabel kan *ikke* repræsentere en prioritetskø, hvor enhver node skal være mindre end alle sine efterkommere.

Denne såkaldte *heap-order* fejler ét sted. I hvilket indeks?

I øverste række er prioriteterne angivet; i nederste række står de tilsvarende tabelindeks.

	14	17	16	28	22	65	29	31	30	26	23	89	64	35	32	48	47	46	45
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Der sker fejl i index 6. Her er parrent større end den child på index 13. Hvilket betyder at den ikke er gyldig.