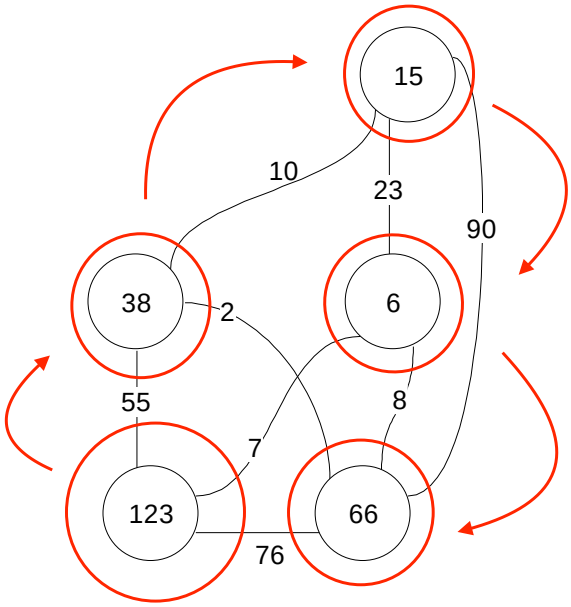
Opgaver onsdag den 2. april

Opgave 1

Betragt nedenstående graf



a) Giv mindst to forskellige eksempler på, i hvilken rækkefølge knuderne kan besøges/udskrives ved et **dybdeførst** gennemløb af grafen med start i knuden 123.

b) Giv mindst to forskellige eksempler på, i hvilken rækkefølge knuderne kan besøges/udskrives ved et **breddeførst** gennemløb af grafen med start i knuden 123.

Opgave 2

I den udlevede klasse *GraphAlgortihm* finder du nedenstående metode signatur

```
/**
 * Returnerer en liste af grafens knuder fundet ved et dybde først gennemløb af
 * grafen med startknude v.
 */
public static <V> List<V> dfs(Graph<V> graph, V v)
```

Programmer metoden og prøv den af på grafen fra opgave 1.
Bemærk: Der kan med fordele laves ekstra hjælpemetoder 😊

Opgave 3

I den udlevede klasse *GraphAlgortihm* finder du nedenstående metode signatur

```
/**
 * Returnerer en liste af grafens knuder fundet ved et bredde først gennemløb af
 * grafen med startknude v.
 */
public static <V> List<V> bfs(Graph<V> graph, V v)
```

Programmer metoden og prøv den af på grafen fra opgave 1.

Opgave 4

Tilføj til klassen *GraphAlgortihm* en metode der givet en graf, kan afgøre om grafen er sammenhængende. Metoden skal tage en graf som parameter og returnere en *boolean.*

Opgave 5

Tilføj til klassen *GraphAlgortihm* en metode der givet en graf og to knuder kan afgøre, om der findes en vej mellem de to knuder.

Opgave 6

a) Find på papiret et letteste udspændende træ for grafen i opgave 1.
b) Find på papiret den korteste afstand til hver af de andre knuder i grafen fra opgave 1, idet der startes i knuden 123.

Opgave 7*

I den udlevede klasse *GraphAlgortihm* finder du nedenstående metode signatur

```
/**
 * Returnerer en mængde af grafens kanter der udgør det letteste udspændende
 * træ for grafen.
 * Grafen er en simpel vægtet graf.
 */
public static <V> Set<Edge> mst(Graph<V> graph)
```

Programmer metoden og prøv den af på grafen fra opgave 1.