	Navn:		Skole:	
	Klasse: 20		Dato: 15. marts 2021	Fag: Matematik A

Opgave 004

Mit klasse diagram, jeg bruger en federe version af JavaScript som hedder TypeScript, det har mulighed for at bruge interfaces og types

```
interface ClasseDiagram {
  generateHeadline(): string;
}

class SportFakeNews implements ClasseDiagram {
  syntaxes: string[] = [];
  categorys: Map<string, string[]> = new Map();

  constructor() {
    this.syntaxes.push(
      "{land} {vinder over | taber til} {land} i {sport}",
      "{sport} udgår nu som OL disciplin",
      "BREAKING: {land} fyrer {person} efter {event} med {person}",
      "{person} stiller op i {sport} til OL",
      "{person} brækker sin {kropsdel} efter {event} mod {person}"
    );

    this.categorys.set('person', ['Cristiano Ronaldo', 'LeBron James', 'Lionel Messi', 'Neymar Jr.', 'Kevin Durant', 'Tiger Woods', 'Rafael Nadal', 'Novak Djokovic', 'Maria Sharapova'])
    this.categorys.set('sport', ['fodbold', 'tennis', 'håndbold', 'volleyball', 'golf', 'svømning', 'basketball', 'hockey', 'cykling']);
    this.categorys.set('land', ['Danmark', 'Tyskland', 'Usa', 'Sverige', 'Norge', 'Rusland', 'Kroatien', 'Syd Afrika', 'Island']);
    this.categorys.set('kropsdel', ['lillefinger', 'ben', 'arm', 'ryg', 'fod', 'storetå', 'albue']);
    this.categorys.set('event', ['stort opgør', 'hemmeligt forhold']);
  }

  generateHeadline(syntax = this.randomSyntax()): string {
    let newString = this.parseSyntax(syntax);

    // Make the headline Title cased
    return this.toTitleCase(newString);
  }

  parseSyntax(syntax: string): string {
    let newString = syntax;
    const matches = this.getArguments(syntax);
```

```
    matches?.forEach((val: string) => {
      const list = val.includes('|') ? val.split('|').map(s => s.trim()) : this.categorys.get(val);
      if (!list) throw new Error(`${val} er ikke en rigtig liste af ord`);
      let sentenceToReplaceWith = list[Math.floor(Math.random() * list.length)];
      if(sentenceToReplaceWith.includes('{')) sentenceToReplaceWith =
this.parseSyntax(sentenceToReplaceWith);

      newString = newString.replace(`{${val}}`, sentenceToReplaceWith);
    });

    return newString;
  }

  private randomSyntax(): string {
    return this.syntaxes[Math.floor(Math.random() * this.syntaxes.length)];
  }

  private toTitleCase(str: string) {
    return str.charAt(0).toUpperCase() + str.substr(1);
  }

  private getArguments(str: string): string[] {
    const matches: string[] = [];
    const bracketStack: number[] = [];

    str.split('').forEach((char, idx) => {
      if(char == '{') bracketStack.push(idx);
      if(char == '}') {
        if(bracketStack.length == 1) {
          const val = str.slice((bracketStack.pop() || 0) + 1, idx);
          matches.push(val);
        } else { bracketStack.pop(); }
      }
    });

    return matches;
  }
}
```