Dokumentation

De funktioner du skriver kommer ofta användas av andra utvecklare. De behöver dokumentation för att förstå hur de ska använda dina funktioner (även du kan behöva dokumentationen när du glömt hur din funktion fungerar)

Följande information brukar ingå i dokumentation:

1. Vad gör funktionen?
2. Vilka input tar funktionen?
3. Vad returnerar funktionen?
4. Ett elller flera exempel som visar på hur funktionen fungerar

Dokumentationsstandard

De flesta språk har en standard för att skriva dokumentation i. Om man följer den standarden kan dokumentationen användas av din texteditor för att ge dig hjälp, eller av ett annat verktyg för att generera en dokumentations-site (som t.ex docs.ruby-lang.org)

TomDoc

I Programmering 1 kommer ni använda er av dokumentationsstandarden TomDoc.

Dokumentationen består av kommentarer som "sitter ihop" med funktionsdefinitionen, och ser ut som följer:

```
# Duplicate some text an arbitrary number of times.
#
# text  - The String to be duplicated.
# count - The Integer number of times to duplicate the text.
#
# Examples
#
#   duplicate("David", 4)
#   # => "DavidDavidDavidDavid"
#
#   duplicate("+-", 6)
#   # => "+-+-+-+-+-+-"
#
# Returns the duplicated String.
def duplicate(text, count)
  i = 0
  output = ""
  while i < count
    output += text
    i += 1
```

```
  end
  return output
end
```

På den första raden ska du kortfattat beskriva vad funktionen gör. Tänk att dokumentationen är till för andra programmerare som kan vilja använda din funktion.

Efter beskrivningen följer en tom rad, sen funktionens argument (i exemplet ovan tar funktionen två argument: `text` och `count`). Beskriv vilken datatyp argumentet ska ha, och vad den representerar.

Efter argumenten kommer en tom rad, sen ordet "Examples". Efter det en tom rad till, sen ett eller flera exempel som visar hur funktionen fungerar vid olika input. Notationen `# =>` visar vad funktionen returnerar med givna argument.

Här är ett exempel till:

```
# Returns the next even Integer following number
#
# number - The Integer that you want the next following even Integer of.
#
# Examples
#
# next_even(3)
# => 4
#
# next_even(6)
# => 8
def next_even(number)
    if number % 2 == 0
        output = number + 2
    else
        output = number + 1
    end
    return output
end
```

Kontroll av output

Om du vill testa att din dokumentation är korrekt formaterad kan du installera följande gems:

`gem install pygmentize` `gem install tomdoc`

kör sen `tomdoc filens-namn.rb` och dokumentationen kommer skrivas ut i kommandoprompten.

```
➜  Desktop tomdoc dup.rb
-------------------------------------------------------------------------------
-----
main#duplicate(text, count)
```

```
Duplicate some text an arbitrary number of times.

text  - The String to be duplicated.
count - The Integer number of times to duplicate the text.

Examples

  duplicate("David", 4)
  # => "DavidDavidDavidDavid"

  duplicate("+-", 6)
  # => "+-+-+-+-+-+-"

Returns the duplicated String.
```