Kalkulator

Skrevet av: Ole Kristian Pedersen, Kodeklubben Trondheim

Oversatt av: Stein Olav Romslo

Kurs: Python
Tema: Tekstbasert
Fag: Matematikk, Programmering
Klassetrinn: 5.-7. klasse, 8.-10. klasse

Kalkulator

I denne oppgåva skal du lage ein kalkulator heilt på eiga hand Det er meininga at du skal skrive all koden sjølv, men du får nokre hint for å hjelpe deg på veg.

Me vil at kalkulatoren skal kunne addere (`+`), subtrahere (`-`), multiplisere (`*`) og dividere (`/`). Me kallar `+` , `-` , `*` og `/` for *operatorar*, og i denne oppgåva skal du lage ein funksjon for kvar operator (du kan til dømes kalle dei `add` , `subtract` , `multiply` og `divide` , som er dei engelske namna på operasjonane operatorane gjer). Kvar funksjon skal ha to tal som parametrar, og skal både utføre rekneoperasjonen og skrive ut svaret.

Brukaren skal sjølv skrive inn kva rekneoperasjon som skal utførast.

Døme på bruk av programmet:

```
>>>
Operator: *
First number: 3
Second number: 5
3 * 5 = 15
>>>
```

Klar, ferdig, programmer!

Då er det berre å setje i gang!

Her er nokre ting å tenke på:

- Korleis avgjer du kva operasjon som skal utførast?
- Har rekkefølgja på tala noko å seie? Er `4-2` lik `2-4` ?

Viss du står fast kan det vere lurt å lese tipsa i dei gule boksane.

int()

Når du får input frå brukaren får du ein tekst, som på fagspråket kallast *streng*, sjølv om brukaren skreiv inn eit tal. Då er det greitt å kunne konvertere teksten til eit tal, ved å bruke `int()`.

Kva er skilnaden på desse kodesnuttane? Du kan køyre koden og teste sjølv.

```
tal = input("Skriv eit tal: ")
svar = 3 + tal
print(svar)
```

```
tal = int(input("Skriv eit tal: "))
svar = 3 + tal
print(svar)
```

Funksjonar med parametrar

Ein funksjon blir deklarert, altså definert, ved hjelp av `def` -nøkkelordet. Den kan brukast ved å skrive funksjonsnamnet med parentesar bak.

Døme:

```
def hello_word():
    print("Hello World!")

hello_world()
```

Ein funksjon som har *parametrar* blir deklarert med parametrar på innsida av parentesane i definisjonen av funksjonen.

Døme:

```
def greet(firstName, lastName):
    print("Hello, " + firstName + " " + lastName)
```

Når me kallar funksjonen seinare, så gir me den *argument*.

Døme:

```
greet("Ola", "Nordmann")
```

Du la kanskje merke til skiljet mellom *parametrar* og *argument*. Ein parameter er namnet me gir variabelen i funksjonsdefinisjonen, slik som `firstName` og `lastName`. Argument er dei verdiane me gir til funksjonen når me kallar den, slik som `"Ola"` og `"Nordmann"`.

Test programmet

- [ ] Fungerer programmet som det skal? Viss ikkje må du rette på det.

Delt på null

- [ ] Kva skjer når du deler på null? Prøv til dømes `4 / 0`.

Viss programmet ditt feilar no, så har du truleg ein delt-på-null-feil. Ein kan nemleg ikkje dele på null! Fiks programmet ditt slik at programmet skriv ut `"Du kan ikkje dele på null!"` viss brukaren prøver å dele på null. Slik:

```
>>>
Operator: /
First number: 4
Second number: 0
Division by zero is not allowed!
>>>
```

Fleire utrekningar

- [ ] Endre programmet ditt slik at brukaren kan skrive inn kor mange utrekningar kalkulatoren skal utføre. Då vil programmet fungere slik:

```
>>>
How many calculations? 4
Operator: +
First number: 0
Second number: 246
0 + 246 = 246

Operator: *
First number: 3
Second number: 255
3 * 255 = 765

Operator: /
First number: 4
Second number: 0
Division by zero is not allowed!

Operator: /
First number: 0
Second number: 4
0 / 4 = 0.0

>>>
```

Fleire operasjonar

- [ ] Prøv å leggje til fleire operatorar. Du kan til dømes leggje til `**` -operatoren. Den opphøger eit tal i eit anna. Til dømes er `2**3` lik `8` fordi `2*2*2` er lik `8` .

- [ ] Kjem du på andre operatorar som kan leggjast til i kalkulatoren din?