Datorövningar i mängdlära

Syfte

Mängder är en vanligt förekommande datatyp i programmeringsspråk. I den här laborationen får du arbeta med mängdoperationer och relationer i Python. Förhoppningsvis ger det också bättre förståelse för hur mängder fungerar rent matematiskt.
Det finns en uppgift (nummer 4) där du får skriva en egen Python-funktion.

Mängder i Python

Det finns en datatyp för mängder i Python. Den kallas `set` och skiljer sig från listor och tupler i och med att elementen inte är ordnade. Därför kan man inte indexera element i en mängd, dvs man kan inte använda uttryck som `m[i]`.

De flesta mängdoperationerna vi går igenom i kursen finns redan definierade i Python. Här är ett urval av de viktigaste:

- `set()` skapar en tom mängd ∅
- `set([`*e1,...,en*`])` skapar en mängd med alla elementen från listan, dvs {*e1,...,en* }. Kan också ta en sträng eller tupel, eller en mängd som helt enkelt kopieras.
- `{` *e1,...,en* `}` skapar också en mängd.
- *x* `in` *s* testar medlemskap ∈
- *x* `not in` *s* testar icke-medlemskap ∉
- *s*`==`*t* testar likhet =
- *s*`.issubset(`*t*`)` eller *s* `<=` *t* testar delmängd ⊆
- *s*`.union(`*t*`)` eller *s* `|` *t* ger unionen ∪
- *s*`.intersection(`*t*`)` eller *s* `&` *t* ger skärningen/snittet ∩
- *s*`.difference(`*t*`)` eller *s* `-` *t* ger differensen \
- *s*`.update(`*t*`)` lägger till alla elementen i *t* till *s*
- *s*`.add(`*x*`)` lägger till *x* till *s*
- *s*`.discard(`*x*`)` tar bort *x* från *s*
- `len(`*s*`)` ger mängdens kardinalitet |*s*|.

Du hittar fler operationer dokumenterade här.

För att kunna ha en mängd som ett element i en annan mängd måste du använda typen `frozenset`. Detta är en mängd från vilken man inte kan ta bort eller lägga till element. Med andra ord, `frozenset` förhåller sig till `set` som `tuple` förhåller sig till `list`. Här är ett exempel:

```
a = set([1,2,frozenset([3])]).
```

Detta motsvarar mängden {1, 2, {3}}.

Uppgifter

Lösningar till samtliga uppgifter ska sparas i en Python-fil. Lägg gärna alla tester i print-uttryck, t ex `print("2 a", 1 in A)`, så det går att se vad som händer när man kör filen. Ledtexten ”2 a” här talar om vilken uppgift det gäller.

1) Det finns inte någon funktion för komplementet. Skriv en sådan funktion. Du kommer också att behöva definiera en variabel som representerar universum U = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.
   Den ska fungera så här:
   ```
   >>> comp({0,1,2,3,4})
    {5,6,7,8,9}
   ```
   Viktigt! Funktionen ska returnera komplementet, inte skriva ut det.

2) Skapa följande mängder: A={1, 2, 3, 4}, B ={3, 4, 5}, C= {5}, och låt U= {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Testa sedan om följande uttryck är sanna, *först* i tanken eller på papper, och *sedan* i Python.
   a) $1 \in A$
   b) $5 \in A$
   c) $1 \in \emptyset$
   d) $1 \in U$
   e) $A \subseteq B$
   f) $C \subseteq B$
   g) $A \subseteq \emptyset$
   h) $A \subseteq U$
   i) $A \cap C \subseteq B$
   j) $A \cup C \subseteq B$
   k) $A \backslash B = C$
   l) $B \backslash C \subseteq A$
   m) $B \backslash C = A$
   n) $A \subseteq C^c$
   o) $|U| = 5$
   p) $|A| + |B| = 5$

3) Testa också dessa (använd frozenset):
   a) $A \in \{ \{2, 1\}, \{2, 1, 3, 4\} \}$
   b) $A \subseteq \{ \{2, 1\}, \{2, 1, 3, 4\} \}$
   c) $C \subseteq \{ \{2, 1\}, \{2, 1, 3, 4\} \}$

4) Det finns ingen funktion för att skapa produktmängden A×B av två mängder A och B. Skriv en sådan funktion. Den ska fungera så här:

```
>>> setprod({1,2,3}, {4,5})
{(1,4),(2,4),(3,4),(1,5),(2,5),(3,5)}
```

Obs! Den ska returnera den nya mängden, inte skriva ut den! Den ska alltså sluta med
något i stil med:

```
    return m
```

Den ska inte innehålla några `print`-satser.

Ett par skapar man i Python helt enkelt med parenteser, t ex

```
    p = (e1,e2)
```

Vill du komma åt de enskilda elementen i en mängd så kan du loopa över mängden, så här:

```
for e1 in m1:
    # Gör något med elementet e1
```

Vill du komma åt de enskilda elementen i *två* mängder så kan du loopa över båda två samtidigt. Det gör du genom att ha den ena loopen *innanför* den andra:

```
for e1 in m1:
    for e2 in m2:
        # Gör något med elementen e1 och e2
```

Du kan skapa en tom mängd m med `m = set()`, och du kan lägga till element p till m med `m.add(p)`.

5) Beräkna/testa följande med din produktmängdsfunktion:
   a) $A \times B$
   b) $A \times (B \times C)$
   c) $A \times B = B \times A$
   d) $A \times \varnothing = \varnothing$

Checklista

[ ] Uppgift 1 till 5 sparade i en fil och redovisade.

[ ] Era namn står i en kommentar i början av filen.