28 juli Programmering, Pepper - Pass 1

1. Vad kommer skrivas ut av följande uttryck? Om du är osäker så testa att skriva in uttrycken i pythonterminalen.

(a) 5.6 - 8
(b) 5,6 - 8
Vad är skillnaden mellan 5.6 och 5,6?

Tilldela x och y värden först, såklart!
(c) x/y
(d) x//y
(e) x%y
(f) x+=4
(g) x++

2. Lite operationer med strängar, fundera först vad som händer, innan ni testar. Reflektera och fråga om nått är oklart.
(a) x = ”Hello world”
(b) x[7]
(c) x[1:3]
(d) x[-2]
(e) x[1:]
(f) x[:] # När vill man använda detta?
(g) x[:1]
(h) x[:-3]
(i) x[-5:]
(j) x*4
(k) x+"!!"

3. Vi ska nu skriva ett program som omvandlar tum och fot till cm. Vi vet att 1 tum = 2.54 cm och 1 fot = 12 tum. Det är vanligt att skriva ” istället för tum och ’ istället för fot. 5’2” är alltså 5 fot och 2 tum. Skapa en funktion som tar in en sträng med ovanstående format och skriver ut längden i cm. Testa med till exempel följande indata:
(a) 5"
(b) 2’6"
(c) 7’3"
(d) 13’
(e) Gör så användaren får välja om det ska konverteras från cm till tum och fot, eller från tum och fot till cm.

4. Vi ska nu skriva rekursiva funktioner, dvs. funktioner som kallas sig själva. Man skulle kunna lösa dessa problem med loopar också men gör inte det nu eftersom vi ska lära oss om rekursiva funktioner.

(a) Skriv en ny funktion som du döper till ”fakultet” med en parameter ”värde”. Om man kallar funktionen fakultet(5) så ska man få 5! men funktionen ska fungera med alla positiva heltal inom rimlig storlek.

(b) Gör en liknande funktion där man får fibonaccitalföljden. Om man kallar funktionen fibonacci(5) så ska man få det 5:e fibonaccitalet men funktionen ska fungera med alla positiva heltal inom rimlig storlek.

5. Vi ska nu använda oss av listor. När vi skriver listor i Python använder vi hakparenteser, [, och skiljer de olika variablerna i listan genom att skriva ett kommatecken. Se exemplet nedanför.

```
frukter = ["Äpple", "Päron", "Banan"]
```

Vi kan därefter använda en for-loop på listan vi har skrivit och gå igenom varje element för sig. Ett exempel på detta är

```
frukter = ["Äpple", "Päron", "Banan"]
for frukt in frukter:
   print(frukt)
```

Som kommer skriva ut varje frukt för sig efter varandra (testa gärna detta innan ni går vidare). I python börjar numreringen av index (positioner) i listor på 0.

Skapa en lista med namn på minst 3 personer. Skriv ett program som skriver ut ”Hej” och därefter namnet på personen för varje person.

6.

Vad händer när man skriver följande? Hur ser a ut efter varje rad? Fundera innan ni skriver in det i terminalen. Ger någon rad error?

```
>>> a = [2,3]
>>> a[1] = 5
>>> a[1] = [3,4,5]
>>> a[2] = [7]
>>> a[2:] = [7]
>>> a[2:] = [7,8,9,10]
>>> a[2:] = [3,4]
>>> a[:-3] = [0]
```

7.

Använd list comprehensions för att lösa följande uppgifter.

(a) Skapa en lista med alla kvadrattal mellan 1 och 100.

(b) Skapa en lista med alla tal mellan 1 och 100 som innehåller siffran 1.

(c) Utgå från listan [”Björn”, ”Hund”, ”Katt”, ”Pedro”]. Lägg till ett utropstecken till alla strängar och filtrera bort alla strängar som inte innehåller bokstaven ’n’.

(d) Vad gör följande kod? Tänk efter innan ni testar koden.

```
[x for x in range(2,100) if not [y for y in range(2,x) if x % y == 0]]
```

Funkar koden bra på större tal? Kan man göra en liten ändring så att koden blir lite snabbare för större tal?

8. Vi använder ”while” när vi inte vet hur många gånger vi ska köra en loop. Ett exempel på detta är

```
tal = input("Ange lösenkod: ")
korrekt = "1337"
while tal != korrekt:
   tal = input("Ange lösenkod: ")
print("Rätt kod! Nu är du i den hemliga delen av programmet!")
```

Där programmet stoppar när vi har skrivit in rätt lösenkod. Testa detta före du fortsätter.
Alla program som kan skrivas med en for-loop kan även skrivas med en while-loop, men inte omvänt.

Skriv ett program som frågar efter ditt namn och låt loopen köra så länge det är fel. När du skriver in rätt namn skriver programmet ut ”korrekt”.

9. Du ska göra en funktion som innehåller en for-loop och en annan funktion som innehåller while-loop. Låt funktionerna beräknade följande, och testa vilken som passar bäst.

(a) Fakultet

(b) Fibonacci

10. Du ska skapa en namnlista av obestämd längd. Ta hjälp av koden nedan och en while-loop. När man skriver in ”end” ska koden skriva ut listan och programmet avslutas.

```
namn = input("Skriv in ett namn: ") # Låter användaren skriva in ett värde (av typ
    string/text) och skriver det till variabeln namn.

namnlista = [] # Skapar en tom lista

namnlista.append(namn) # Lägger till ett element i slutet av listan, i det här fallet värdet
    av variabeln namn.
```

11. Dictionaries är en variant på listor fast den sparar nyckel-värde par istället. Nyckelarna måste vara unika (vad betyder unik i detta sammanhang? Är ”a” och ”A” unika? Är 1 och 1.0 unika?).

```
color_of_fruits = {
"apple": "red",
"pear": "green",
"banana": "yellow"
}
print(color_of_fruits["apple"])
```

Om du vill kan du kopiera exempelkoden och testa.

(a) Skriv en dictionary med namn som indexering och som har ålder som värde.

För att byta värdet till ett av elementen kan man skriva

```
color_of_fruits["apple"] = "green"
```

(b) En av personerna i din dictionary har precis fyllt år, och du vill därför ändra åldern på personen. Skriv in ändringen i programmet.

För att lägga till ett element i en dictionary skriver man

```
color_of_fruits["pineapple"] = "yellow"
```

(c) Lägg till en person i din dictionary med en ålder.

Att skriva dict() är sätt att skapa en tom dictionary. Precis som list() skapar en tom lista. Det går också bra att skriva {} respektive [ ].

12. Förutom listor och dictionaries kan man även använda sig utav något som heter tuples. Tuples är som listor, men skiljer sig på ett sätt. Ni ska nu själva få upptäcka hur dom skiljer sig. Ett exempel på en tuple är:

```
months = ('January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
    'September', 'October', 'November', 'December')
```

(a) Skriv en tuple som innehåller olika färger. Skriv därefter ut första och tredje elementet.

För att lägga till ett element i en lista skriver man

```
listans_namn.append(<elementet man vill sätta in>)
```

medan man i dictionaries skriver

```
dictionary_namn["index"] = <elementet man vill sätta in>
```

(b) Försök att lägga till ett element i din tuple med färger.

(c) Ge några exempel på när tuples kan vara bra att använda.

Extrauppgifter

13. Skriv olika funktioner som känner igen olika intervall med hjälp av b.la. if-satser.

(a) Funktion som tar in våglängd och ger färg.

(b) Funktion som tar in frekvens och ger ton.

Ni kan använda Wikipedias sidor för att kolla färgernas intervall i nm och tonernas namn i Hz.