Regler

- Student får lämna salen tidigast en timme efter tentans start.
- Vid toalettbesök eller rökpaus ska pauslista utanför salen fyllas i.
- All form av kontakt mellan studenter under tentans gång är strängt förbjuden.
- Böcker och anteckningssidor kan komma att granskas av tentavakt i samband med tentans start samt under tentans gång.
- Frågor om specifika uppgifter eller om tentan i stort ska ställas via tentasystemet.
- Systemfrågor kan ställas till assistent i sal.
- Ingen uppgift rättas efter tentatidens slut.
- Ingen kompletteringsmöjlighet ges de sista tio minuterna.
- Så länge en uppgift inte har betyget U kan den kompletteras till godkänt.
- Inga elektroniska hjälpmedel får medtas. Mobiltelefon ska vara avstängd och ligga i jacka eller väska.
- Inga ytterkläder eller väskor får förvaras vid skrivplatsen.

Antal uppgifter	5
Antal sidor (exklusive denna)	5
Hjälpmedel	En pythonbok (tex Learning Python 4 ed.) Ett A4-ark med egna anteckningar

VÄND!

Betygssättning

Tentan innehåller fem uppgifter. För godkänt betyg krävs minst två avklarade uppgifter. För gränser för högre betyg, se i tabell 1.

Tid (timmar)	Antal uppgifter	Betyg
3	3	5
4	4	5
4	3	4

Tabell 1: Betygssättning

Bonus från labserien

Bonus från labserien gäller endast vid förstagångstentamen och går därför inte att använda vid detta tillfälle.

Information

Inloggning

Logga in på ditt normala IDA-konto men välj session “exam system”.

Följ menyvalen så långt det går tills du ska mata in ett engångslösenord. Tag fram ditt LiU-kort och visa det för tentavakten för att få detta lösenord.

Avslutning

Efter att du loggat ut som vanligt ska du vänta ett tag och sedan trycka på knappen “Avsluta tentamen” när det är möjligt. När detta är gjort är det omöjligt att logga in igen.

Uppgift 1

I filen `given_files/bly-i-fisk.txt` finns en tabell som visar hur mycket bly som finns i sill och strömming fiskad utanför Sveriges kust. Värdena är angivna i mikrogram per gram torrvikt och kommer från Naturvårdsverket.

Skriv ett program som läser in tabellen i filen till en lämplig datastruktur. Därefter ska programmet skriva ut en lista med vilken mätplats som fått högst resultat för respektive år samt detta mätvärde. Därefter ska ytterligare en lista skrivas ut som presenterar hur många gånger respektive mätplats haft det högsta värdet sorterade efter antal. Programmet ska ge följande utskrift (så klart med samtliga resultat istället för ...):

Körexempel 1

```
1981: 0.197 (Harufjärden)
1982: 0.141 (Landsort)
    ...
2010: 0.068 (Utlängan)
2011: 0.073 (Väderöarna)
Väderöarna: 1
Fladen: 2
Harufjärden: 3
Ängskärsklubb: 5
Landsort: 8
Utlängan: 12
```

Mätdatat kommer från: `http://www.naturvardsverket.se/Sa-mar-miljon/Statistik-A-O/Bly-i-fisk/`

Uppgift 2

En lysdiod är en elektronisk komponent som har två egenheter, den släpper endast igenom ström i en riktning och den lyser i en given färg. Det finns lysdioder som har flera delar som kan kombineras för att ge en blandning av färger.

Det finns många olika sätt att beskriva färger och en av de vanligaste är RGB-skalan. RGB är en additativ modell som uttrycks med tre tal som representerar styrkan av rött, grönt respektive blått ljus som måste blandas för att få den givna färgen. Dessa tre tal kan presenteras på olika sätt, antingen som ett absolut värde mellan 0 och ett bestämt maxvärde (ofta 255) eller som andel av ett maxvärde uttryckt som ett tal mellan 0 och 1.0. Ett vanligt sätt att uttrycka ett RGB-värde är som ett hexadecimalt tal med sex tecken (två för varje färg). Se tabell 2 för exempel på dessa olika sätt att ange värden.

Färg	Absolut värde	Andel	HEX
Röd	(255, 0, 0)	(1.0, 0, 0)	FF0000
Blå	(0, 0, 255)	(0, 0, 1.0)	0000FF
Vit	(255, 255, 255)	(1.0, 1.0, 1.0)	FFFFFF
Svart	(0, 0, 0)	(0, 0, 0)	000000
Orange	(255, 128, 0)	(1.0, 0.31, 0)	FF8000

Tabell 2: Olika färger i RGB-skalan

Antag att du har en lysdiod med tre delar; röd, grön och blå. Varje del kan ges allt mellan 0 och 5 volt för att variera ljusstyrkan på aktuell färg från avslaget till full styrka. Skapa ett program som ber användaren om en färg uttryckt hexadecimalt och skriver ut vilken spänning respektive färg behöver för att lysdioden ska lysa med den inmatade färgen enligt nedanstående körexempel. Spänning ska skrivas ut med två decimaler.

Körexempel 1

```
Mata in färg: FF0E00
Röd:  5.00V
Grön: 0.27V
Blå:  0.00V
```

Körexempel 2

```
Mata in färg: FF8000
Röd:  5.00V
Grön: 2.51V
Blå:  0.00V
```

Uppgift 3

I spelet Boggle slumpas sexton bokstäver ut i ett kvadratiskt rutmönster och sedan ska spelarna försöka skapa så många ord som möjligt av de givna bokstäverna. I denna uppgift ska vi spela en förenklad variant av Boggle.

Antag att vi har ett språk som har definitionen att ett ord är fyra tecken med exakt två vokaler. Din uppgift är att hitta så många ord som möjligt från ett givet Bogglespel enligt föregående definition. Orden får endast tas radvis, kolumnvis eller efter diagonalerna (till skilnad från riktiga Boggle där man kan kombinera tecken på många fler sätt). Orden kan läsas från båda håll på brädet (exempelvis radvis både från vänster och höger sida). Användaren matar in bokstäverna från Boggle-brädet som en rad med sexton tecken. Ditt program ska sedan skriva ut alla unika ord på det givna brädet samt antal unika ord. Ordningen på orden i utskriften spelar ingen roll.

Du kan anta att de fyra första tecknen motsvarar första raden på brädet, de fyra nästa andra raden och så vidare. Se nedanstående exempel:

```
Användarinmatning: ABCDEFGHIJKLMNOP
Faktiskt spelbräde:
  A B C D
  E F G H
  I J K L
  M N O P
```

Körexempel 1

```
Spelbräde: ABECDGZFIÅCMNYRÖ
Hittade unika ord (totalt 12):
YÅGB
CEBA
AGCÖ
MCÅI
BGÅY
ÖRYN
ÖCGA
NIDA
IÅCM
ADIN
ABEC
NYRÖ
```

Körexempel 2

```
Spelbräde: ABBABAABBAABABBA
Hittade unika ord (totalt 2):
ABBA
BAAB
```

OBS: För er som inte minns det är tecknen A, E, I, O, U, Y, Å, Ä och Ö definierade som vokaler.

Uppgift 4

I denna uppgift ska du implementera en (väldigt utrymmesmässigt krävande) krypteringsalogritm given nedan.

```
Indata: Ett medddelande M med N tecken

Steg 1: Skriv ut talet N som ett tecken
Steg 2: För varje tecken C i M:
  Steg 2.1: Skriv ut C
  Steg 2.2: Slumpa ett tal I i intervallet [1, 95]
  Steg 2.3: Skriv ut talet I som ett tecken
  Steg 2.4: Skriv ut I stycken slumpmässiga tecken
Steg 3: Upprepa steg 2.2 till 2.4 en extra gång
```

Varje gång du ska omvandla från ett tal till ett tecken adderar du 31 till talet och tar ut tecknet på denna positionen i teckentabellen. Detta för att få ett skrivbart tecken (tecknen på position 0 till 31 är så kallade styrtecken).

Antag att du har meddelandet “Hej”. Då ska följande hända:

```
Längd: 3
1: tecknet på position (3+32) => " skrivs ut
2.1: H skrivs ut
2.2: Antag att vi slumpar talet 4
2.3: tecknet på position (4+32) => # skrivs ut
2.4: 4 slumpmässiga tecken, exempelvis f2G/ skrivs ut
```

Detta upprepas sedan för resten av meddelandet.

Körexempel 1

```
Meddelande: Hejsan
Krypterat: &H+%rS.*n(^y^[e$2n3pj!Ls+C:=Uhx8Up*paGQg$K,5`B3(Sy$lAQ)Y/V(r4L
ZaRXuFzds1?d:/1nFgnM)EmEuhYELn&e_`J.hG<SL#E"`1)j$QV"cX6wOD1WEH%%PZHZk-''p
',pV#&ghj(_Itf+|WUMp[>87f0X5#{2kC!fT]Wu3NmB>'~j"pHqfR.B3q6Q:+/6kw<d]z@
```

OBS: Då krypteringen bygger på slump är det inte troligt att ditt program får exakt samma utskrift som i exemplet.
TIPS: Funktionerna `chr` och `ord` kan vara bra att ha för att jobba med teckentabellen:

```
>>> chr(35)
'#'
>>> ord('a')
97
```

Uppgift 5

Skapa en högre-ordningens funktion `check_passwords(users, password_check)` som tar en tabell `users` innehållandes användarnamn (nyckel) och deras ännu icke-krypterade lösenord (värde). Funktionen ska använda funktionen `password_check` för att testa om lösenorden uppfyller givna krav på ett lösenord. De användarnamn som inte uppfyller testet ska returneras i en lista som resultat. Om alla lyckas returneras alltså en tom lista. Gör sedan två funktioner som använder sig av `check_passwords` och även dem returnerar användare med felaktiga lösenord på samma sätt som `check_passwords`:

```
check_passwords_classic(users):
       Lösenordet ska vara minst 6 tecken långt och innehålla
       minst 2 siffror och minst 2 bokstäver.

check_passwords_sloppy(users):
       Lösenordet ska vara minst 3 tecken långt och innehålla
       minst 3 olika tecken.
```

I filen `given_files/passwords.py` finns funktionerna deklarerade samt ett kort testprogram som testar era funktioner. Kopiera denna fil till din hemkatalog och gör ändringar i den. Körexemplet nedan visar vad testprogrammet bör visa när du har en fungerande lösning (dock kan användarnamnen komma i en annan ordning eftersom vi använder tabeller).

Körexempel 1

```
Användare med ogiltiga lösenord: Kalle, Jonny, Stina
Användare med väldigt dåliga lösenord: Jonny, Stina
```