2008-01-14

Tentamen för TDP002 Imperativ programmering - problemlösningsdel

Uppgifterna löses enskilt vid IP-salarnas datorer med Ubuntu på egen hårddisk. Skriftligt material, Internet och egna filer tillåtna. Kommunikation med utomstående person/annan deltagare (direkt eller via utbyte av filer) strikt förbjudet.

Betyg 3: 7 poäng.
Betyg 4: 11 poäng.
Betyg 5: 14 poäng
Totalt: 20 poäng (5 uppgifter)

Det totala betyget på tentamen är detsamma som det lägsta betyget av teoridel och problemdel.

Tid för problemlösningsdel: 2 timmar. Invänta nästa del i anslutning till tentasalen (enligt tentavaktens order).

Skicka in svar via Urkund

Skicka in uppgifterna en i taget som bifogad fil till följande epostadress:

tobiv@ida.liu.se

Svaren kommer vidarebefodras och processas av anti-fusk-systemet **Urkund**.

I samtliga uppgifter uppmuntras ni att skriva lättläst källkod och införa underfunktioner vid behov.

Uppgifter

Uppgift 1 (3p):

a) Skriv en funktion `format_time(seconds)` som gör en tid angiven i sekunder till en tupel med fyra element: dagar, timmar, minuter och sekunder. . T ex ska argumentet `73` ge som resultat tupeln `(0,0,1,13)`.

b) Skriv en funktion add_format_time(time1, time2) som adderar två tal representerade i tupelformat enligt a) . T ex argument (0,2,52,44) och (1,0,17,23) bli tupeln (1, 3, 10, 7)

Uppgift 2 (3p): Skapa ett program `number_file.py` som frågar användaren efter namnet på en in-fil och en ut-fil i konsolen, laddar in angiven in-fil och skriver ut filen på angiven ut-fil. Om infilen inte fins eller utfilen inte kan skapas ska felmeddelande skrivas ut i konsolen. Utfilen ska eka ut innehållet på in-filen men här radnummer och antal kolumner i början av varje rad. Kolumnen inom parentes och kolon innan orginalraden börjar. Tomma rader (inkl rader med mellanslag, tabtecken etc) ska inte ha radnummer. T ex för en infil med detta innehåll:

```
aa
bbb

cccc
```

så ska utfilen se ut så här:

```
1(2): aa
2(3): bbb

3(4): cccc
```

Uppgift 3 (4p): Gör en högre-ordningens funktion `check_passwords(password_check, dict)` som tar en dictionary som par av användarnamn och deras ännu icke-krypterade lösenord. Funktionen ska använda funktionen password_check för att testa om lösenorden uppfyller kraven på ett lösenord. De användarnamn som inte uppfyller testet ska returneras i en lista som resultat. Om alla lyckas returneras en tom lista. Gör sedan två metoder som använder sig av check_passwords och också returnerar felaktiga lösenord på samma sätt som check_passwords:

`check_passwords_classic(dict)`: lösenordet måste vara minst 6 tecken långt och bara innehålla bokstäver och siffror, med minst två siffror och minst två bokstäver.

`check_passwords_sloppy(dict)`: lösenordet ska vara minst 3 tecken långt och måste innehålla minst 3 olika tecken

Vi antar att åäö inte förekommer i användarnamn och lösenord (varken i korrekta eller felaktiga)

Uppgift 4 (5p): Skapa ett program make_properties.py som översätter konfigureringsfiler baserat på det klassiskt Unix-vis till Pythons ini-format som används på Windows med sektioner. Använd modulen ConfigParser för att generera den nya filen. Ange namnet på infil (enklare unix-format) och utfil (Python format). Det enklare formatet antas vara på en rak lista av med tilldelningar på formen egenskap=värde. Man har använt namnkonvention med punkt för att hantera liknande namn, vilket ska översättas till sektioner. Exempel på infil:

```
global.length=23
global.width=17
global.area.color=red
fallback.color=blue
fallback.time=23
```

Utfilen ska då bli:

```
[global]
length=23
width=17
[global.area]
color = red
[fallback]
color=blue
time=23
```

Uppgift 5 (5p): Skapa en modul shopping.py som hanterar en kundvagn för Internet-shopping. Du väljer själv den interna representationen för kundvagnen. Följande funktioner ska ingå:

create_shopping_cart(person): skapar en kundvagn för en person med givet namn

add_product(name, quantity price, cart): lägg till produkter till kundvagnen. Kundvagnen ska registrera produktnamn, antal och pris.

lookup_shopping_cart(person, cart_list): hitta och returnera kundvagnen för en person med givet namn i en lista av kundvagnar.

check_out(cart, delivery_cost): Summerar och returnera den totala kostnaden för kundvagnen och lägger till given fraktkostnad.

print_shopping_cart(cart): skriver ut kundvagnens innehåll i konsolen. I utskriften ska ingå: namnet på personen, varornas namn, antal och styckpris samt totalkostnaden för hela kundvagnen