HÖGSKOLAN I HALMSTAD

Tentamen i programmering DI2006 2020-03-16

Tid: 9:00-13:00

Lärare: Johan Thunberg, 035-167273

Anvisningar

- Inga hjälpmedel tillåts (förutom penna, suddgummi och **bifogad appendix**).
- Vid varje uppgift anges hur många poäng som maximalt utdelas för uppgiften.
- Denna tentamen består av två delar: **Del 1** och **Del 2**. På Del1 kan man få maximalt 30 poäng. På Del 2 kan man få maximalt 20 poäng. Försök att uttrycka er kortfattat i svaren i Del 1.

- För betyg 3 gäller det att få minst 15 poäng på Del1 (Del 2 behöver alltså inte göras för att få detta betyg).
- För betyg 4 gäller det att få minst 15 poäng på Del1 och minst 8 poäng på Del 2.
- För betyg 5 gäller det att få minst 15 poäng på Del1 och minst 13 poäng på Del 2.

Lycka till allesammans!

Del 1 (Denna del består av sex uppgifter)

Uppgift 1 (4p)

Betrakta följande utryck:

a) `1+(2%1)`
b) `1+2-(3*4/3)%2//1`
c) `(1 > 2) or (2 < 1)`
d) `not (True or False) and (”hej” == ”hopp”)`

Ange värdet för varje uttryck ovan.

Uppgift 2 (6p)

a) Vad kommer variabeln `value` ha för värde efter respektive loop A)-D)?

```
value = 0
for i in range(3):
    value += i**2
```

A)

```
value = 0
for i in range(0,10,2):
    value += i/4
```

B)

```
value = 1
while value < 5:
    value = value**2 + 1
```

C)

```
big_value = 8
value = 1
while big_value - value > 1:
    big_value -= 1
    value     += 1
```

D)

b) Betrakta koden nedan. Ersätt `pass` inuti for-loopen med lämplig kod så att utskriften på skärmen blir: 123456789

```
string = ""
for i in range(10):
    pass

print(string)
```

Uppgift 3 (5p)

a) Betrakta funktionen `foo` nedan. Om ett värde returneras av `foo`, vad blir detta värde för respektive val av input:

a = [1,2], b = (1,2), c = "12", d = "1", e = range(3)

```
def foo(x):
    value = x[0]
    for i in x[1:]:
        value += i
    return value
```

b) Skriv en funktion som tar som argument ett positivt heltal (dvs ett tal större än 0). Funktionen använder sedan funktionen `foo` för att räkna ut summan av alla tal från 0 till det positiva talet i fråga. Summan returneras sen av funktionen. För att få full poäng på denna deluppgift krävs någon typ av felhantering som tar hand om fallet när input till funktionen inte är ett positivt heltal.

Uppgift 4 (5p)

Betrakta de två funktionerna `foo` och `bar`.

```
def foo(arg):
    f = open(arg,"w")
    for i in range(10):
        for j in range(10):
            f.write(str(random.randint(0,1)))
        f.write("\n")
    f.close()

def bar(arg):
    nr_ones = 0
    # ...
    return nr_ones
```

a) Vad gör funktionen `foo`? Beskriv övergripande och inte rad för rad.

b) Ersätt ”# …” i funktionen `bar` med lämplig kod så att funktionen returnerar antalet förekomster av tecknet ”1” i en textfil med namnet `arg`.

Uppgift 5 (5p)

Betrakta följande program:

```
def foo(text):
    var1 = text.split()
    var2 = set(var1)
    var3 = {}
    for i in var2:
        var3[i] = 0
        for j in var1:
            if i == j:
                var3[i] += 1
    return var3


var = foo("hej hej på dig!")
for i in var:
    print(i + ": " + str(var[i]))
```

a) Ange vad `var1`, `var2`, respektive `var3` är för typ av objekt (om `foo` är en sträng).
b) Ange (inte rad för rad utan övergripande) vad funktionen `foo` gör.
c) Vad skrivs ut på skärmen när programmet körs?

Uppgift 6 (5p)

Betrakta exemplet nedan där en klass återfinns som representerar (tvådimensionella) bollar. Ange alla rader där det förekommer fel (logiska, syntaktiska osv). För varje rad, skriv något mycket kortfattat om hur felet eller felen kan korrigeras.

```
1 class Ball
2     def __init__(self, x_val,y_val,r_val):
3         x = x_val
4         y = y_val
5         r = r_val
6
7     def get_all(self):
8         return x,y,r
9
10    def __str__(self):
11        return self.x + "," \
12    self.y + ',' + self.r
13
14 ball = ball()
15 values = ball.get_all()
```

Del 2 (denna del består av två uppgifter: 7 och 8)

Uppgift 7. (10p)

Denna uppgift innehåller två deluppgifter. Varje deluppgift ger max 5p.

Om ni t.ex finner uppgift a) svårare än uppgift b) så kan ni börja med uppgift b) som kan göras oberoende av a).

a) Programmera en metod

`generate_key(text)`

som hittar alla unika tecken i strängen `text` och associerar varje sådant tecken med ett unikt tal med hjälp av en dictionary. Denna dictionary returneras av funktionen. Alla tal ska innehålla samma antal siffror.

Talen 101, 943, 301, 402 har samma antal siffror, men talen 1, 98, 101, 77 har inte samma antal siffor. Om `text = "hej hej"` så finns det fyra unika tecken i `text` (inkluderat mellanslaget). I detta fall skapas således en dictionary där nycklarna är dessa tecken och värdena är fyra unika och slumpmässigt valda tal med samma antal siffror.

b) Programmera en metod

`encrypt(string, key)`

som returnerar en krypterad version av strängen `string` där varje tecken `T` i `string` har bytts mot talet (representerat som en sträng) som är värdet för nyckeln `T` i dictionary `key`.

Uppgift 8. (10p)

Skapa en klass `Roulette` som modellerar en förenklad version av hasardspelet Roulette, där en kula skickas iväg i ett roterande hjul med olika numrerade och färglagda fack. Kulan hamnar tillslut i ett fack.

I denna version gäller att:

- finns det 37 olika fack, numrerade mellan 0 till 36,
- varje fack är antigen grönt, rött eller svart. Fack 0 är grönt, medan hälften av de andra facken är svarta och andra hälften är röda. Detta kan för enkelhetens skull väljas slumpmässigt.
- Spelare kan läggas till eller tas bort.
- Innan ett spel börjar så satsar respektive spelare en summa pengar (som inte får överstiga ett belopp som du definierar) på ett av alternativen nedan. Det kan t.ex. vara att satsa på ett nummer, två nummer eller en färg. För varje alternativ nedan så

anges hur många gånger det satsade beloppet som erhålls vid vinst. Vid förlust så förlorar spelaren hela det satsade beloppet. Om kulan hamnar i facket med nummer 0, så förlorar alltid spelaren/spelarna.

- Alternativ: (ett nummer, 35ggr), (två nummer, 17ggr), (tre nummer, 11ggr), (fyra nummer, 8ggr), (sex nummer, 5ggr), (tolv nummer, 2ggr), (arton nummer, 1ggr), (färg, 1ggr).

Klassen ska innehålla metoder för att:

- ta reda på vilken färg ett nummer har,
- lägga till eller ta bort spelare,
- ange hur mycket en spelare satsar och på vad denne satsar på enligt alternativen ovan,
- simulera ett spel och räkna ut hur mycket respektive spelare vann,
- ta bort alla satsningarna för spelarna efter avslutat spel.

Apendix till tentamina i Programmering DI2006
Python: sammanfattning

Kod	**Förklaring**

`name = "Kalle"`
`age = 10`
`pi = 3.14`
`savings= BankAccount()`
	En variabel av **typen string**
	En variabel av **typen int**
	En variabel av **typen float**
	En variabel av typen BankAccount

`print(y)` `print("Hej")`	**Skriver ut** värdet av variabeln y. Skriver ut "Hej"
`x = input("Namn: ")`	**Skriver ut**: "Namn: " och **pausar tills användaren skriver in** något och trycker på Enter. **Lagrar sedan** texten användaren skrev in, i variabeln x.

`x = 15`
`if x < 10:`
`  s = "x < 10"`
`  print(s)`
`elif x < 20:`
`  s = "x < 20"`
`  print(s)`
`elif x == 20:`
`  print("x = 20")`
`else:`
`  print("else:")`
`  print("x > 20")`
	Villkorssatser.
	
	if testar om ett villkor är uppfyllt.
	
	elif hänger ihop med if och testas bara om testet i if misslyckas.
	
	else hänger också ihop med if och testas bara om det i if och eventuella elif misslyckas.

`age == 100`	Exempel på villkor man kan testa. **Kom ihåg: == INTE =** när man vill testa ett villkor!
`x < 100 and y < 100`	Exempel på villkor man kan testa. Testar om både variabeln x **och** variabeln y är mindre är 100.

for tal **in** range(100): print(tal)	**for loop** För varje tal från 0 till och med 99: Skriv ut talet.
for num **in** [1, 6, 2, 5]: print(num)	För varje nummer i listan [1, 6, 2, 5]: Skriv ut numret.

var = 10
while var < 100:
  print(var)
  var = var + 1
	while loop
	Medans villkoret
	var < 100 är uppfyllt, gör följande om och om igen:
	- Skriv ut värdet av var
	- Öka var med ett .

def funktionsnamn(a, b):
  r = a*b + 10
  r = r/10
  r = r – 10
  return r
	Funktionsdefinition. Denna kod körs först då funktionen anropas. a och b kallas för **parametrar** och är variabler utan värde som kan användas inuti funktionen.

funktionsnamn(1, 2) x = funktionsnamn(2, 5)	**Anrop** av funktionen *funktionsnamn* kör koden i funktionsdefinitionen. **Argumenten** 1 och 2 är värden som lagras i parametrarna i funktionsdefinitionen

fil=open("filnamn", "r")
 	
for ord in fil:
  //gör något
 	
text=fil.read()
line=fil.readline()
	Öppna en fil och läser den ord för ord. w- skriv , a-append
	
	läser en fil till variabel text
	
	läser en rad från fil till variabel line

lista=[] for var in lista	Skapa tom lista Iterera genom lista

Skapa klasser och objekt

```
class BankAccount:

     def __init__(self, iname, ibalance):
          self.__name=iname
          self.__balance=ibalance

     def deposit(self, amount):
          balance=balance-amount
```

```
savigs= BankAcount(”kalle”, 100)    // skapa objekt
```

Användbara funktioner:

```
int(a)  // koverterar a från String till int
str(a)  // konverterar a från int till String
ord(a)  // konverterar a från char till int
chr(a)  // konverterar a från int till char

random.randint(a,b) // slumpar tal mellan a och b

text.split(delimitator)  Dela en sträng i en lista av ord.
Argumenten delimitator används som separator. Metoden returnerar
en lista.
```

Andra String metoder:

Table 8-1 Some string testing methods

Method	Description
`isalnum()`	Returns true if the string contains only alphabetic letters or digits and is at least one character in length. Returns false otherwise.
`isalpha()`	Returns true if the string contains only alphabetic letters and is at least one character in length. Returns false otherwise.
`isdigit()`	Returns true if the string contains only numeric digits and is at least one character in length. Returns false otherwise.
`islower()`	Returns true if all of the alphabetic letters in the string are lowercase, and the string contains at least one alphabetic letter. Returns false otherwise.
`isspace()`	Returns true if the string contains only whitespace characters and is at least one character in length. Returns false otherwise. (Whitespace characters are spaces, newlines (`\n`), and tabs (`\t`).
`isupper()`	Returns true if all of the alphabetic letters in the string are uppercase, and the string contains at least one alphabetic letter. Returns false otherwise.

Table 8-3 Search and replace methods

Method	Description
`endswith(substring)`	The *`substring`* argument is a string. The method returns true if the string ends with *`substring`*.
`find(substring)`	The *`substring`* argument is a string. The method returns the lowest index in the string where *`substring`* is found. If *`substring`* is not found, the method returns –1.
`replace(old, new)`	The *`old`* and *`new`* arguments are both strings. The method returns a copy of the string with all instances of *`old`* replaced by *`new`*.
`startswith(substring)`	The *`substring`* argument is a string. The method returns true if the string starts with *`substring`*.

List metoder

Table 7-1 A few of the list methods

Method	Description
`append(item)`	Adds *item* to the end of the list.
`index(item)`	Returns the index of the first element whose value is equal to item. A `ValueError` exception is raised if item is not found in the list.
`insert(index, item)`	Inserts *item* into the list at the specified *index*. When an item is inserted into a list, the list is expanded in size to accommodate the new item. The item that was previously at the specified index, and all the items after it, are shifted by one position toward the end of the list. No exceptions will occur if you specify an invalid index. If you specify an index beyond the end of the list, the item will be added to the end of the list. If you use a negative index that specifies an invalid position, the item will be inserted at the beginning of the list.
`sort()`	Sorts the items in the list so they appear in ascending order (from the lowest value to the highest value).
`remove(item)`	Removes the first occurrence of *item* from the list. A `ValueError` exception is raised if item is not found in the list.
`reverse()`	Reverses the order of the items in the list.

Dictionary metoder:

Table 9-1 Some of the dictionary methods

Method	Description
`clear`	Clears the contents of a dictionary.
`get`	Gets the value associated with a specified key. If the key is not found, the method does not raise an exception. Instead, it returns a default value.
`items`	Returns all the keys in a dictionary and their associated values as a sequence of tuples.
`keys`	Returns all the keys in a dictionary as a sequence of tuples.
`pop`	Returns the value associated with a specified key and removes that key-value pair from the dictionary. If the key is not found, the method returns a default value.
`popitem`	Returns a randomly selected key-value pair as a tuple from the dictionary and removes that key-value pair from the dictionary.
`values`	Returns all the values in the dictionary as a sequence of tuples.