Examen Sistemas de Negocio.

1...

[1. A]

> Count the number of items in a list with the result in a dictionary. List example: l = [’a’, ’b’, ’f’, ’f’, ’b’, ’b’], Should give something like: c = {’a’: 1, ’b’: 3, ’f’: 2}

```
c = {}
l = ['a','b','f','f','b','b']
print "lista = {}".format(l)
for x in l:
    c[x] = l.count(x)
print "conteo = {}".format(c)
```

[1. B]

> Write a short python script to read the contents of a file, consisting of two columns of numbers. Store each line of numbers as pairs of integers in a list. Finally convert this list to a multidimensional numpy array.

```
  import numpy as np
  with open('binum.txt') as f:
      lines = [[int(x) for x in line.split()] for line in f]
  multiarray = np.array(lines)
  print multiarray
```

2...

[2. A]

> Within a python script (that can be run at the command line) create an class object Employee, which has three attributes in its constructor (first_name, last_name, staff_number), and one member function “GetEmployee” member function that returns a string concatenating first_name, last_name, and “staff_number”.

```
class Employee( object ):
    def __init__( self, first_name, last_name, staff_number
        self.first_name = first_name
        self.last_name = last_name
        self.staff_number = staff_number
    def GetEmployee( self ):
        return "This is: '{}, {}' with Staff Number: {}."\
        .format(self.first_name,self.last_name,self.staff_nu
```

[2. B]

> Using your two objects, Create a list, E, consisting of 2 Employees where the staff_number is generated randomly.

```
import random

def RandStaffNum():
    return random.randint(0,100)
```

```
E = []
a = Employee("Daniel", "Camba Lamas", RandStaffNum())
E.append(a)
b = Employee("Pepe", "Chiringo", RandStaffNum())
E.append(b)

for x in E:
    print x.GetEmployee()
```

3...

[3. A]

> Write the commands needed to create a Django “project” within a virtualenv.

```
# Usando tu shell favorito, en tu carpeta de desarrollo...
virtualenv djangoEnv
source ./djangoEnv/bin/activate
pip install django
django-admin.py startproject SN_Exam
```

[3. B]

> Write the command to create a Django “application”.

```
python2 manage.py startapp
```

[3. C]

> Write the data model “Products” within the Django application “Product_Catalog” (specify which file you must modify, and include the following attributes: product_name, product_description, purchase_price, sales_price).

```
python2 manage.py startapp Product_Catalog
vim Product_Catalog/models.py
```

```
from django.db import models
from django.core.validators import MinValueValidator

class Product(models.Model):
    product_name        = models.CharField(max_length = 30)
    product_description = models.CharField(max_length = 200)
    sales_price         = models.FloatField(validators=[MinVa
    purchase_price      = models.FloatField(validators=[MinVa
    updated = models.DateTimeField(auto_now_add=True, auto_no
    timestamp = models.DateTimeField(auto_now_add=True, auto_
```

4...

[4. A]

> What command would you use at the command line to create a new database? Given the file, “insdata.sql”, consisting of a table and 1000 lines of SQL insert lines for this table, how would you insert these lines into postgresql using the command line?

```
sudo su - postgres
createdb SN_Exam
```

```
psql -d SN_Exam -f ins-data.sql
```

[4. B]

> Imagine that you connect to the data model “Products” from problem 3c. Write a python script that contains a function using psycopg2 to “insert” a new element and another function to “select” a product based upon the a string that represents the product_description.

```
sudo su - postgres
createdb SN_Exam_ProductStore
createuser -s -P cambalamas
```

```
import psycopg2

conn = psycopg2.connect("dbname=SN_Exam_ProductStore user=ca
cur = conn.cursor()

def toInsert(pName, pDesc, sPrice, pPrice):
    cur.execute("INSERT INTO product \
    (product_name, product_description, sales_price, purchas
    VALUES (%s, %s, %f, %f)", (pName, pDesc, sPrice, pPrice)
    conn.commit()

def toSelect(keyword):
    sql = "SELECT * FROM product WHERE product_description L
    cur.execute(sql)
    cur.fetchall()
```

5...

> Answer the following related to Programming with SQL: Imagine that you have the two tables: CREATE TABLE users (id int PRIMARY KEY, name varchar(256)); CREATE TABLE address (id_user int, address text); INSERT INTO users VALUES (1, David'); and the address data of this user is given by two inserts: INSERT INTO address VALUES (1, 'Works during winter in Ourense'); INSERT INTO address VALUES (1, 'Works during summers in Boston'); Write a trigger (and appropriate function) that will delete all associated addresses of a user if this user is deleted.

```sql
/* DEFINICION TRIGGER. (Casi seguro...) */

-- Funcion
CREATE OR REPLACE FUNCTION del_user_related()
BEGIN
  DELETE FROM address WHERE id_user = OLD.id
END;

-- Trigger
CREATE TRIGGER nueva_venta
AFTER DELETE ON users
FOR EACH ROW
EXECUTE PROCEDURE del_user_related();
```

6... [X]

> In lab, we studied an eCommerce product catalog for an album product. Imagine that you have a mongoDB consisting of entries similar to the following:
>
> ```
> {
>   type: "Audio Album",
> ```

> ```
>   title: "A Love Supreme",
>   description: "by John Coltrane",
>   details: {
>       artist: "John Coltrane",
>       genre: [ "Jazz" ]
>   },
> }
> ```
>
> [[Write a python interface with pyMongDB that will find the CDs by artist and genre.]]

```
pass
```

7...

[7. A]

> Write a short python script using sklearn to compare the prediction of a Random Forest classifier to an SVM classifier in the context of supervised learning.

```
from sklearn import datasets
from sklearn import metrics

from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import cross_val_predict

model1 = SVC()
model2 = RandomForestClassifier()
# load dataset
iris = datasets.load_iris()
```

```
# calculate accuracy
predicted1 = cross_val_predict(model1, iris.data, iris.targe
predicted2 = cross_val_predict(model2, iris.data, iris.targe
# print accuracy
print "SVM         = {}".format(metrics.accuracy_score(iris.
print "RAND_FOREST = {}".format(metrics.accuracy_score(iris.
```

[7. B]

> How could you implement a 10-fold Cross validation.

```
import numpy as np
from random import randint
from sklearn.model_selection import KFold

# generate model with more than 10 items.
dataset = [x*x for x in range( randint(10,25) )]

# k-fold with k=10.
kf = KFold(n_splits=10)
for train, test in kf.split(dataset):
    print("%s %s" % (train, test))
```

8... [X]

> Imagine you have a huge data set consisting of feature vectors with a large number of attributes. You have used supervised learning to classify new data but you want to confirm “visually” whether your predictions make sense. [[In this context, explain how the tSNE manifold method could be a useful tool (hint: first explain what tSNE is and what it can do, then explain how it could be used here)]]

```
pass
```

9...

> Basic empty Module in Odoo. Explain how you would install an empty odoo module. (hint: You should use the “scaffold” command from odoo to create an empty project). Where should you deposit the directory for your custom project? Explain what you would need to do to deploy this custom module so that it installed.

```
# Donde odoo esté instalado...
odoo.py scaffold SN_Modulo_Examen MisModulos
cp -R MisModulos/SN_Moulo_Examen /usr/lib/python2.7/dist-pac
/etc/init.d/odoo restart

# Por último, vía web accedemos a nuestro Odoo y en la pesta
# tras acctualizar, podremos ver nuestro modulo en el listad
```

10... [X]

> Imagine that you have want to modify the default “Manufacturing Order” workflow in Odoo in order to adapt to the requirements of your company. In particular, you want the final approval of the manufacture order to depend upon the Supervisor if the cost value of raw materials passes a certain amount (5000Euros). [[Explain “qualitatively” how this could be accomplished using the Odoo custom workflow editor. An example of the workflow is given below]].

```
pass
```