Documentation

Installation:

hash.py needs to be inside miniDB directory

Functionality:

Create Hash Table:

Description:
Creates hash table and assigns it to specified object(hash_obj)
Command:
hash_obj = Hash(db_name , table_name , column_name , size , method , save)

db_name:	type: string	|	The name of the database
table_name:	type: string	|	The name of the table
column_name:	type: string	|	The name of the column
size:	type: int	|	Size of the hash table
method:	type: method	|	hashing method to use
save:	type: bool , default: True	|	Determines whether the object is saved

(Hashing methods : hash_division , hash_folding)

Accesss existing hash table:

Description:
Assigns existing hash table and to specified object (hash_obj)
Command:
hash_obj = Hash.existing(db_name , table_name , column_name)

db_name:	type: string	|	The name of the database
table_name:	type: string	|	The name of the table
column_name:	type: string	|	The name of the column

Hash Search:

Description:
Searches for value in hash table and returns record
Command:
hash_obj.search(value)

value:	type: Any	|	The value to search for on the table

Hash Join:

Description:
Performs hash join on two tables , returns and saves joined table
Command:
Hash.join(db_name , tableA_name , tableB_name , column_name ,size , method , final_name)

db_name:	type: string	|	The name of the database
tableA_name:	type: string	|	The name of the table
tableB_name:	type: string	|	The name of the table
column_name:	type: string	|	Name of column the join is performed on
size:	type: int	|	Size of the hash tables that are created
method:	type: method	|	hashing method to use
final_name:	type: string	|	Name of the joined table

(Hashing methods : hash_division , hash_folding)

Visualize Hash Table:

Description:
Creates and shows a visualization of the hash table
Command:
hash_obj.visualize()

Examples:

Let’s assume there is a Database named **“unipi”** that has two tables : **“teachers”** and **“subjects”**

*Table **“teachers”**:*

```
>>> db.select('teachers','*')

## teachers ##
  id (int) #PK#  name (str)    subject (str)      salary (int)
---------------  ------------  ---------------  --------------
              0  Theodoridis    SDBD                       1200
              1  Birbou         Allhlepidrash               700
              2  Patsakis       Python                     1000
              3  Alepis         OOP                         900
              4  Sapounakis     Analysh                    1100
              5  Douligeris     Oures                       850
              6  Metaxiotis     Pliroforiaka                700
              7  Apostolou      AI                         1150
              8  Pikrakis       Metaglwtistes               900
              9  Tsikouras      Algebra                     950
```

*Table **“subjects”**:*

```
>>> db.select('subjects','*')

## subjects ##
subject (str) #PK#    difficulty (str)      semester (int)
--------------------  ------------------  ----------------
Allhlepidrash         Easy                               5
SDBD                  Medium                             5
Analysh               Hard                               1
Algebra               Medium                             1
OOP                   Easy                               4
Metaglwtistes         Easy                               3
```

We will create a hash table object for the table **“teachers”** on the column **“id”** using the division method:

```
>>> hash_obj = Hash('unipi','teachers','id',6,hash_division)
dbdata/unipi_db/teachers.pkl
Loaded "unipi".
Loaded "unipi".
Επιτυχής δημιουργία hash file για τον πίνακα :teachers της βάσης: unipi στην στήλη: id
Hash File Path :dbdata/unipi_db/teachers_hash_on_id.pkl
```

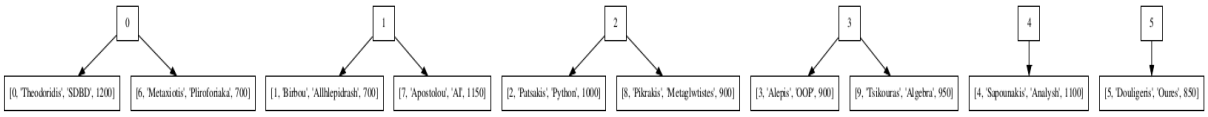
Now, using our hash table we will perform a search on the column **id** for the value **4**:

```
Hash File Path :dbdata/unipi_db/teachers_hash_on_id.pkl
>>> hash_obj.search(4)
[4, 'Sapounakis', 'Analysh', 1100]
>>>
```

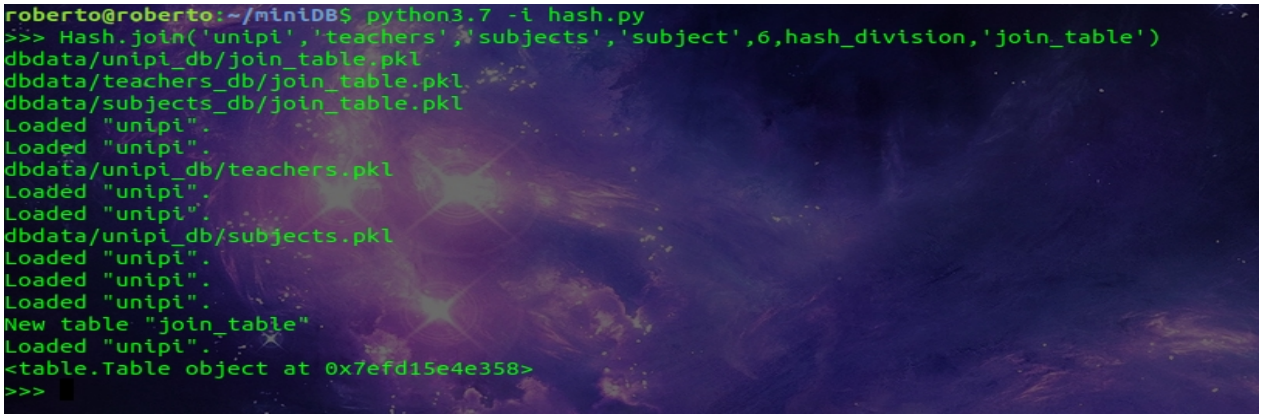
Since we didnt change the **“save”** parameter which is **True** by default , the hash table was saved in the Database directory and we can access it later using the *Hash.existing* method. Now lets visualize the hash table using the *visualize* method:

```
>>> hash_obj = Hash.existing('unipi','teachers','id')
>>> hash_obj.visualize()
>>>
```

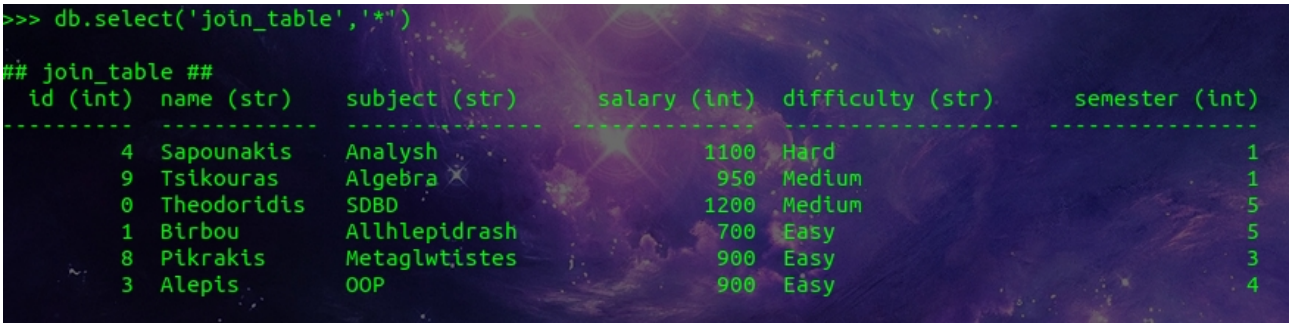
The above renders and outputs the following graph:



We will create a new joined table which joins the tables **“teachers”** and **“subjects”** on the common column **“subject”** and we will name it **“join_table”**:



Finally, let’s view the table:



Credits:

-Νικόλαος Χελιώτης | ΑΜ: Π18172
-Ρομπέρτο Ράπεσης | ΑΜ: Π18131