FilesConstructor

Order.Object

"""
tuple: date = (int: dd, int: mm, int: yy)
dd = 1 ~ 31; mm = 1 ~ 12; yy = 0 ~ 99
"""
Constructor (tuple: <date>,
list: < list: <
str: <IrlenPre>,
None / list: <1EPre>,
None / list: <2EPre>,
None / dict: <DeliveryAddress>,
None / dict: <PaymentMethods>,
None / list: <Extras>,
None / str: <Report on improvement> > >)
ret Order.o

Interface

- **add_pre** (str: <IrlenPre>,
  None / list: <1EPre>,
  None / list: <2EPre>,
  None / dict: <DeliveryAddress>,
  None / dict: <PaymentMethods>,
  None / list: <Extras>,
  None / str: <Report on improvement>)
- **remove_pre** (Pre.o)
  ret (bool: <True / False>)
- **get** (),
  ret (tuple: <date>,
  list: < tuple: <
  Pre.o,
  Pre.**get** () > >)
- **len**()
  ret (int: <number of prescriptios>)
- **date** ()
  ret (tuple: <date>)
- **update_date** (tuple: <date>)
  ret (bool: <True / False>)

1

OrdersManager.Object

Constructor ()

ret Ordermanager.o

Interface

Orders:

- **list_orders** ()
  ret (list: <Order.**get**()> # The list is sorted)
- **list_date_orders**()
  ret (list: <date> # The list is sorted)
- **create_orders_report** (str: <file> # The path and name of the file,
  str: <mode> # "r" – range, "l" – list,
  # in mode "r":
  tuple: <
  str: <from date>,
  srt: <to date> >)
  # in mode "l":
  list: <date>)
  ret (bool: <True / False>)

Order:

- **add_order**(tuple: <date>,
  list: < list: <
  str: <IrlenPre>,
  None / list: <1EPre>,
  None / list: <2EPre> >)
  ret (bool: <True / False>)
- **remove_order**(str: <date>)
  ret (bool: <True / False>)
- **update_date** (tuple: <new date>,
  tuple: <old date>)
  ret (bool: <True / False>)
- **search_order** (tuple: <date>)
  ret (tuple: < Order.**get**(), date > / None)
- **get_last_order** ()
  ret (tuple: < Order.**get**(), date > # "last order" /
  / None # " If order list is empty")

Prescription

- **add_pre** (tuple: <date>,
  str: <IrlenPre>,
  None / list: <1EPre>,
  None / list: <2EPre>)
  ret (bool: <True / False> / Exception)
- **remove_pre** (tuple: <date>,
  Pre.o)
  ret (str: < "remove Pre" / "remove order" # if Order.len() = 1 > /
  "order not found" / "Pre not found")

Customer.Object

"""
Dict: profile stract:
Keys:
"First name"
"Last name"
"Parent name"
"Phone"
'City'
'Street'
'Building number'
'Apartment'
'ZIP code'
"Email"
"Referred by"
"Reason for referral"
Valu type of all the keys is **string**
"""

Constructor (dict: <profile>)

ret Customer.o

Interface

- **get_profile**()
  ret (dict: <profile>)
- **update_profile**(dict: <profile>)
  ret (Nune)
- **get_order_manager**()
  ret (OrderManager.o)

ShulamitApp

SecurityManager.Object

Constructor ()

ret SecurityManager.o

Interface

- **list_customers_managers** (str: < Developer Key>)
  ret (list: <username> / None # If the Developer Key is wrong)
- **add_ customers_manager** (str: <username>,
  str: <password> / None)
  ret (CustomerManager.o / None # If the username already exists)
- **get_customers_manager** (str: <username>,
  str: <password>)
  ret (CustomerManager.o / None # If the customers manager does not exist or the password is wrong)
- **remove_customers_manager**(str: <username>
  str: <password> / None)
  ret (str: < "success" / "failed" # If the customers manager does not exist or the password is wrong>)
- **data_backup** (str: < Developer Key>,
  str: <path>,
  str: <file_name # The full name of the file >)
  ret (str: <"success" / "ErrorKey" # If the Developer Key is wrong / "ErrorPath" # If the specified path does not exist >)
- **data_recovery** (str: < Developer Key>,
  str: <file # The path and name of the file >)
  ret (str: < "success" / "ErrorFile" # If file was not found / "ErrorKey" # If the Developer Key is wrong >)
- **security_update_customers_manager** (str: <username>,
  str: <password> / None,
  str: <new username>,
  str: <new password> / None >)
  ret (str: < "success" / "ErrorUsername" # If the new username already exists / "failed" # If the username or if the password is wrong >)

CustomersManager.Object

"""
Str: customer name = Cust.o.profile["First name"] + " " + Cust.o.profile["Last name"]
"""

Constructor ()

ret CustomerManager.o

Interface

Customers

- **list_customers** ()
  ret (list: < str: <customer name> # The list is sorted >)

Customer

- **search_cust** (str: <customer name>)
  ret (bool: <True / False>)
- **add_cust** (dict: <profile>)
  ret (str: < customer name / None # If the customer name already exists >)
- **get_profile_cust** (str: <customer name>)
  ret (dict: <profile> / None # If the customer name does not exist)
- **remove_cust** (str: <customer name>)
  ret (bool: <True / False> / str: "remove old customer Error!" # If the customer name has changed and the deletion of the old customer name was not successful)
- **update_profile_cust** (str: <customer name>,
  dict: <new profile>)
  ret (bool: <True / False>)
- **get_order_manager_cust** (str: <customer name>)
  ret (OrderManager.o / None # If the customer name does not exist)

Prescription.Object

"""
Prescription type format:
    IrlenPre type: str
    EPre1/2 type: list [PrescriptionData, DistanceOD, DistanceOS, AddOD, AddOS] or Nune
        All value types are: dict
    DeliveryAddress type: dict with the keys are 'City', 'Street', 'Building number', 'Apartment', 'ZIP code'
        All value types are: str or None
    PaymentMethods type: str
        Values: 'PayPal', 'Credit Card'
    Extras type: list
         All value types are: str
    Report on improvement type: str

EPre:
    PrescriptionData type: dict with the keys are: 'Patient Name', 'Expiration Date', 'Pupillary Distance', 'Prescribed by'
        All value types are: str or None
    DistanceOD, DistanceOS, AddOD, AddOS type: dict with the keys are: 'Sphere', 'Cylinder', 'Axis', 'Prism', 'Base'
        All value types are: str or None

EPre structure:
    PrescriptionData = {'Patient Name': None, 'Expiration Date':None, 'Pupillary Distance': None, 'Prescribed by': None}
    DistanceOD = {'Sphere': None, 'Cylinder': None, 'Axis': None, 'Prism': None, 'Base': None}
    DistanceOS = {'Sphere': None, 'Cylinder': None, 'Axis': None, 'Prism': None, 'Base': None}
    AddOD = {'Sphere': None, 'Cylinder': None, 'Axis': None, 'Prism': None, 'Base': None}
    AddOS = {'Sphere': None, 'Cylinder': None, 'Axis': None, 'Prism': None, 'Base': None}
"""

Constructor (
str: <IrlenPre>,
None / list: <1EPre>,
None / list: <2EPre>,
None / dict: <DeliveryAddress>,
None / dict: <PaymentMethods>,
None / list: <Extras>,
None / str: <Report on improvement)

ret Prescription.o

Interface

- **get** (),
  ret (tuple:<
  str: <IrlenPre>,
  None / list: <1EPre>,
  None / list: <2EPre>,
  None / dict: <DeliveryAddress>,
  None / dict: <PaymentMethods>,
  None / list: <Extras>,
  None / str: <Report on improvement> >)