**Contents**

* 1. *General info*
  2. *Item class*
  3. *Category class*
  4. *Product class*
  5. *Service class*
  6. *Sorting Algorithm*
  7. *Main function*
  8. *Other noteworthy things*
  9. **General info**

This program is designed to store items of the commercial shop in a well-sorted manner. There are classes which will create the model of the item in our database. It also includes functions to determine price, effective rate and effective price of the product. Methods are present which will return name and other members of the class.

* 1. **Item class**

**Virtual double getTotalPrice():** This virtual function will ensure that program overrides it to get total price of items. Thus it’s value is zero (0).

**Virtual string getName():** Virtual function to get the name member of the instances.

* 1. **Category class**

This class is used to define category of the products. It has private member textKey and other public members like description, name and parent\_cat.

* 1. **Product class**

**Product class**: has private textKey member because textKey shouldn't be accessed from the other classes. It also has basic data slots for name of the product, price, discount if possible, etc.

**Product()**: There is a default Product constructor.

**Product(string nam, string dscrptn, string textK, double prce, int quant, double dscnt, discountType a, Category c):** Constructor taking all arguments as well. Constructor handles with the negative inputs throwing error.

**getEffectivePrice**(): calculates the price after discount. It also checks if the discount is bigger than price, in which case throws an error, thus ensuring that methods don't return negative result.

**getTotalPrice():** calculates the total price which is ***quantity times effective price.***

**1.5 Service class**

**Class Service**: Service class inherits Product class and has protected own members that can't be accessed from the other classes. It also has basic data slots for name of the product, price, discount if possible, etc.

**Service():** There is a default Service constructor

**Service(string nam, double dur, double ratee, double rateDscnt, discountType a):** Constructor taking all arguments as well. Constructor handles the negative inputs throwing error.

**getEffectiveRate():** calculates the rate after discount. It also checks if the **rateDiscount** is bigger than the rate in which case throws an error, thus ensuring that methods don't return negative result.

**getTotalPrice():** calculates the total price which is ***total price of item plus duration times effective rate.***

* 1. **Sorting Algorithm – Merge Sort**

Merge sort is used to sort out the total prices of all the items.

**merge():** this function is the driver of the merge sort. It takes an array, and 3 other integer variables (beginning, middle, last indexes of an array) as an argument. Then it creates two temporary arrays which will store elements of an array to be sorted. The main operation of this function: it compares the two elements and if one of them is bigger than the other, it swaps their positions. The function ensures that no any element is left in the temp arrays in the end

**swap():** it mainly changes the positions of two variables by storing value of one of them in the temporary variable.

**mergeSort():** this recursive function runs until the array is completely divided into sub arrays, compared and merged back as a sorted array.

**printArray():** the name speaks for itself; it prints elements of an array in a loop with the **printf()** function.

**delete array:** delete function is called because we have to free memory after every temporary array is created and used.

* 1. **Main function**

**Vector<Item\*> items:** stores Item instances

**vector<Category\*> categories:** to store Category instances

**The other fields are instantiation of objects and storing them in the respective vectors.**

For loop prints out name and price of every item in the vector.

**1.7 Other noteworthy things**

**enum discountType {amount, percentage }:** declare type of the discount. It works for both Service and Product class.