

Homework 5

- C++ Environment for grading is **Dev-C++ 5.11** (C++ Compiler: **TDM-GCC 4.9.2**)
 - Please ensure that your code files **can be compiled correctly** before submitting on YZU Portal. Otherwise, there will be **0 points** for that task.
 - Submissions that are **more than 90% similar** will be **reduced 5% increasingly** based on the submission time.
 - All of the tasks in this homework require using **Object-Oriented Programming** and **Templates** (if needs) in C++.
 - Penalty for late parts: **-10% of value for each day late**.
-

Task 1 (20 points): Using **C++ templates** to solve the following exercises:

- Find min, max between 2 elements of type T (**int, double** and **fraction**).
- Find the smallest negative element in an array of type T (**int, double** and **fraction**).
- Find the largest positive element in an array of type T (**int, double** and **fraction**).
- Sort ascending array of type T (**int, double** and **fraction**).

*Note: Students should reuse **Fraction** class from exercise from overloading Homework.*

Task 2 (20 points): There are 4 sorting requirements on an array of numbers as follows:

- Sort the array of numbers in ascending order.
- Sort the array of numbers in descending order of absolute value.
- Sort even numbers ascending and odd numbers descending.
- Sort negative numbers descending, positive numbers ascending, and negative numbers come before positive numbers.

*Note: Students **must** use **C++ Templates** for this task.*

Task 3 (30 points): A company manages the purchasing of 3 types of customers:

Type A: Ordinary customers (including x customers):

Payment = Quantity of goods * Unit price + VAT (10%)

Type B: Loyal customers (including y customers):

Promotion percentage = MAX (Number of years of loyalty * 5%, 50%)

Payment = (Number of goods * Unit price) * (100% – Promotion percentage) + VAT (10%)

Type C: Special customers (including z customers):

Payment = (Number of goods * Unit price) * 50% + VAT (10%)

Write a program that does the following:

- Import customer list from `customer.imp` text file. $0 < x, y, z < 1000$.
- Use the function to calculate the **Total Payment** of each customer in the payment list. Export the results to the file `payment.out` in the below format.
- Design and build classes to solve the problem of calculating the total amount of money earned by the company. Export the results to the end of the `payment.out` file.

customer.imp	Samples	payment.out
x y z NameA1 NumberOfGoodsA1 UnitPriceA1 ... NameAx NumberOfGoodsAx UnitPriceAx NameB1 NumberOfGoodsB1 UnitPriceB1 YearsOfLoyaltyB1 ... NameBy NumberOfGoodsBy UnitPriceBy YearsOfLoyaltyBy NameC1 NumberOfGoodsC1 UnitPriceC1 ... NameCz NumberOfGoodsCz UnitPriceCz	3 1 2 Chia-hao 30 1000 Chih-ming 50 3000 Chun-chieh 10 5000 Chien-hung 10 1500 30 Chih-hao 60 2000 Chih-wei 20 3500	x y z NameA1 PaymentA1 NameAx PaymentAx NameB1 PaymentB1 NameBy PaymentBy NameC1 PaymentC1 NameCz PaymentCz TotalPayment

Task 4 (30 points): A hotel in Taipei has 3 types of rooms with monthly room revenue calculated as follows:

Deluxe type: room revenue = (Number of nights * TWD\$7500 + Service fee) * 115% + Extra fee

Premium type: room revenue = (Number of nights * TWD\$5000 + Service fee) * 105%

Business type: room revenue = Number of nights * TWD\$3000

Write a program that does the following:

- Import the list of rooms used from the `rooms.imp` text file.
d: number of rooms of Deluxe type
p: number of rooms of Premium type
b: number of rooms of Business class
 $(0 < d, p, b < 1000)$
- Define a function to calculate revenue for each type of hotel room. Output the results to `revenue.out` file in the below format.
- Design and build classes to list rooms with excellent revenue. Knowing the room with excellent revenue meets the following properties:
 - Monthly revenue \geq Last month's revenue * 125%
 - Output the results at the end of the `revenue.out` file.

rooms.imp	Last month	Current month	revenue.out
d p b	2 3 4	2 3 4	d p b
RoomCodeD1	deluxe01	deluxe01	RoomCodeD1
ServiceFeeD1	3000	6000	RevenueD1
ExtraFeeD1	4000	4000	...
NumberOfNightsD1	3	6	RoomCodeDd
...	deluxe02	deluxe02	RevenueDd
RoomCodeDd	5000	5000	RoomCodeP1
ServiceFeeDd	2000	2000	RevenueP1
ExtraFeeDd	5	5	...
NumberOfNightsDd	premium01	premium01	RoomCodePp
RoomCodeP1	7000	7000	RevenuePp
ServiceFeeP1	7	7	RoomCodeB1
NumberOfNightsP1	premium02	premium02	RevenueB1
...	10000	10000	...
RoomCodePp	2	4	RoomCodeBb
ServiceFeePp	premium03	premium03	RevenueBb
NumberOfNightsPp	20000	20000	=====
RoomCodeB1	10	5	Excellent Room:
NumberOfNightsB1	business01	business01	RoomCode1
...	4	8	RoomCode2
RoomCodeBb	business02	business02	RoomCode3
NumberOfNightsBb	6	5	...
	business03	business03	
	15	10	
	business04	business04	
	14	20	