CPT105 Introduction to Programming in Java 2021-2022 S1
Resit Coursework Task Sheet

Overview

Resit coursework consists of three parts: Part A, B, and C.

The learning outcomes of CPT105 Intro to Programming in Java include equipping you to understand and appreciate the principles of Object-Oriented Programming. We also want you to be competent to design, write, test and debug programs in Java. Part A and Part B are designed to start you in those directions, where you will write your own classes and methods.

In addition, you are also expected to be aware of the design and the good documentation of a finished program. Part C will help you achieve that goal.

The resit Coursework is 100% of your final grade – it is an individual project.

In addition, we may ask you to give a short presentation/Q&A to show the authenticity of your project.

Resit Coursework – Part A

In Part A, you are required to write a GUI program to simulate data encapsulation when we transfer data via the Internet. You should follow the description to produce the correct output. It is further divided into two parts: Part A.1 and A.2.

CW3 Part A will contribute to 35% of the total marks.

Part A.1 DataPacket (12 marks)

We usually transfer encrypted data packets via the Internet to protect the privacy of original data. Usually, an encrypted data packet is composed of several components: the frame header, frame tail, length of data and encrypted data. Table 1 below shows the detailed packet structure used in the question.

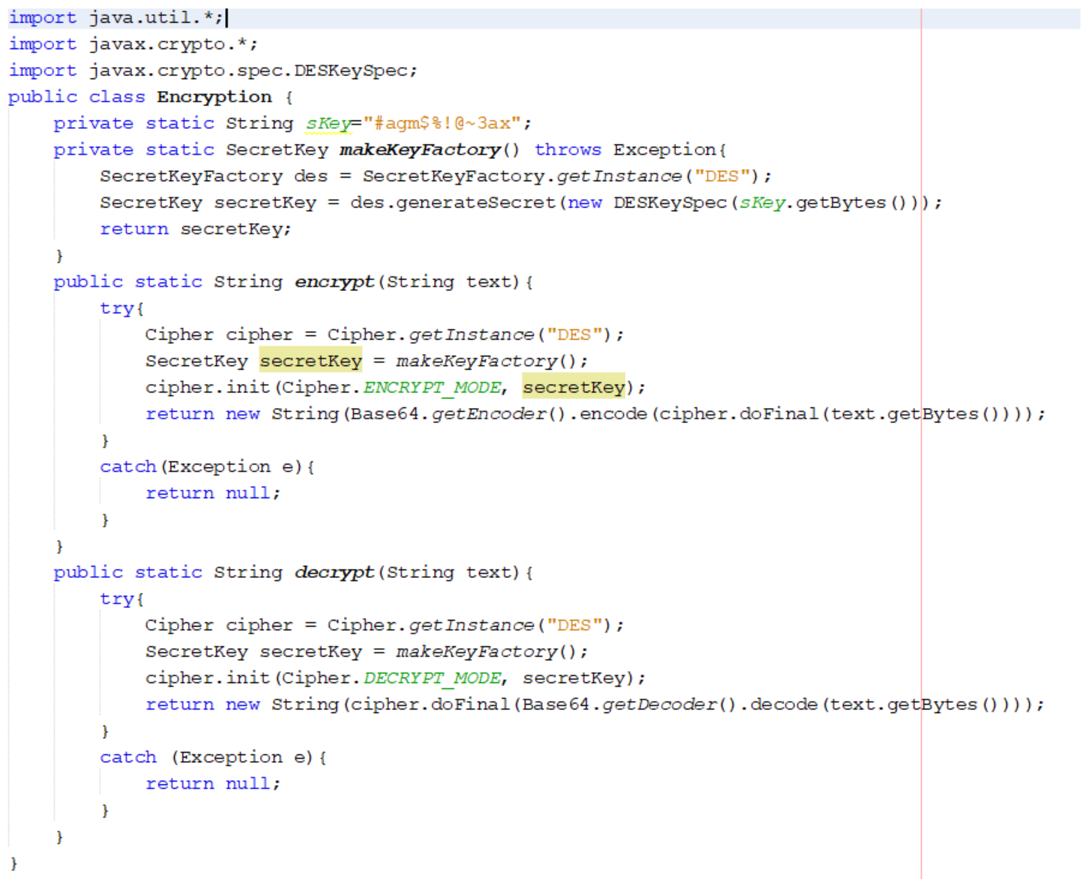
Table 1. Packet structure

Header	Data length	Encrypted data segment	Tail
2 Bytes	1 Byte	N Bytes	2 Bytes

In Table 1, we use 2 hexadecimal constants of "AAAA" and "BBBB" to represent the header and tail of a data packet (Header and Tail columns), respectively, one byte to represent the length of the encrypted data. We should encrypt the original data and encapsulate the encrypted data into the data segment and convert the data packet to a hexadecimal string before we send the data via the Internet. The procedure of the data encapsulation is as follows:

(1) Encrypt the data string into an encrypted string.
(2) Calculate the length of the encrypted string.
(3) Convert encrypted string into a hexadecimal value.
(4) Combine the header, data length, encrypted data and tail into a data packet.

For example, before we send a string "123" to a remote user via the Internet. The string should be encrypted and encapsulated into a data packet of a hexadecimal string "AAAA0c44464567552f507a6e4f513dBBBB". "AAAA" and "BBBB" are the header and tail of the data packet, "0c" represents the length of encrypted data (12 bytes), "44464567552f507a6e4f513d" is the hexadecimal string of each character in the encrypted data "DFEgU/PznOQ=". We defined a class "Encryption" to help you encrypt or decrypt strings as follows:



```java
import java.util.*;
import javax.crypto.*;
import javax.crypto.spec.DESKeySpec;
public class Encryption {
    private static String sKey="#agm$%!@~3ax";
    private static SecretKey makeKeyFactory() throws Exception{
        SecretKeyFactory des = SecretKeyFactory.getInstance("DES");
        SecretKey secretKey = des.generateSecret(new DESKeySpec(sKey.getBytes()));
        return secretKey;
    }
    public static String encrypt(String text){
        try{
            Cipher cipher = Cipher.getInstance("DES");
            SecretKey secretKey = makeKeyFactory();
            cipher.init(Cipher.ENCRYPT_MODE, secretKey);
            return new String(Base64.getEncoder().encode(cipher.doFinal(text.getBytes())));
        }
        catch(Exception e){
            return null;
        }
    }
    public static String decrypt(String text){
        try{
            Cipher cipher = Cipher.getInstance("DES");
            SecretKey secretKey = makeKeyFactory();
            cipher.init(Cipher.DECRYPT_MODE, secretKey);
            return new String(cipher.doFinal(Base64.getDecoder().decode(text.getBytes())));
        }
        catch (Exception e){
            return null;
        }
    }
}
```

Please follow the steps below to complete the task:

(1) Create an empty java project. The project name should be "Resit_1".

(2) Create a class "Encryption" in your project.

  (a) Copy the code from the figure above into your Encryption class;

  (b) Change the encryption key to your student id.

(3) Create a class "DataPacket".

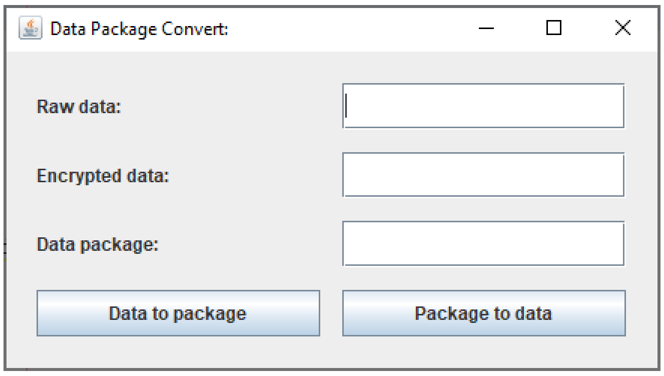
  (a) Define a method "strToHex(String str)" to convert a common string into a hexadecimal string. For example, convert a string "123A" to "31323341".

  (b) Define a method "convertToDataPacket(String data)", which can encrypt a string into a new encrypted string and encapsulate it into a data packet and return a hexadecimal string, as Table 1 shows. For example, encrypt a string "123" into a data packet string "AAAA0c 44464567552f507a6e4f513dBBBB".

  (c) Define a method "getDataFromDataPacket(String hexData)", which can retrieve the original data from a hexadecimal string of a data packet. The method should check the header and tail of the packet and the length of the data. For example, retrieve the original data string "123" from the data packet "AAAA0c44464567552f507a6e4f5 13dBBBB"

(4) Create a class "Q1" and achieve the following functions in its main method.

  (a) Create a frame window as the following;



(b) Create a DataPacket object in the main method;

(c) When users click "Data to packet" button, the string in the raw data text field can be encrypted and showed in the encrypted data text field. The encrypted data can be encapsulated into a data packet string, and the data packet is shown in the data packet text field;

(d) When users click "Packet to data" button, it can retrieve the encrypted and original data strings from the data packet field and show them in the raw data and encrypted data text fields, respectively.

Part A.2 MultiDataPacket (23 marks)

Sometimes, we need to transfer several data within one data packet to improve data transmission efficiency. For example, if we send three strings of "ABC", "EFG" and "HIJ" in one data packet, the hexadecimal value of the data packet is "AAAA0c685279554936796637696b3dBBBBAAAA0c61427a616c76474133366b3dBBBBAAAA0c5861346a4970775755756b3dBBBB". The sub string of "AAAA0c685279554936796637696b3dBBBB" is the hexadecimal data of the first encrypted string "ABC". The sub strings of "AAAA0c61427a616c76474133366b3dBBBB" and "AAAA0c5861346a4970775755756b3dBBBB" are the hexadecimal data for encrypted strings of "EFG" and "HIJ", respectively. In addition, you should check the validation of the data packet before we retrieve data from it. If the data packet is not valid, you should throw an exception. Please follow the steps below to complete the task.

(1) Create an exception class "MyException", which is inherited from "Exception".

  (a) Create a constructor "MyException (String message)" to create an exception object with a string of the exception message.

  (b) Override the method "toString()" to return the message string.

(2) Create a subclass "MultiDataPacket" based on the base class "DataPacket".

  (a) Create an overloading method "String[] strToHex(String[] str)", which can convert an array of common strings to an array of hexadecimal strings.

  (b) Create an overloading method "String convertToHexDataPacket (String[] data)", which can convert an array of common strings to a MultiDataPacket with a single hexadecimal string.

(c) Create a new method "String[] getMultiDataFromHexDataPacket (String hexData)" to retrieve and return an array of strings from the MultiDataPacket string "hexData". The method should validate the header, tail and length for each data packet in the string "hexData" and throw a "MyException" exception if any validation is failed.

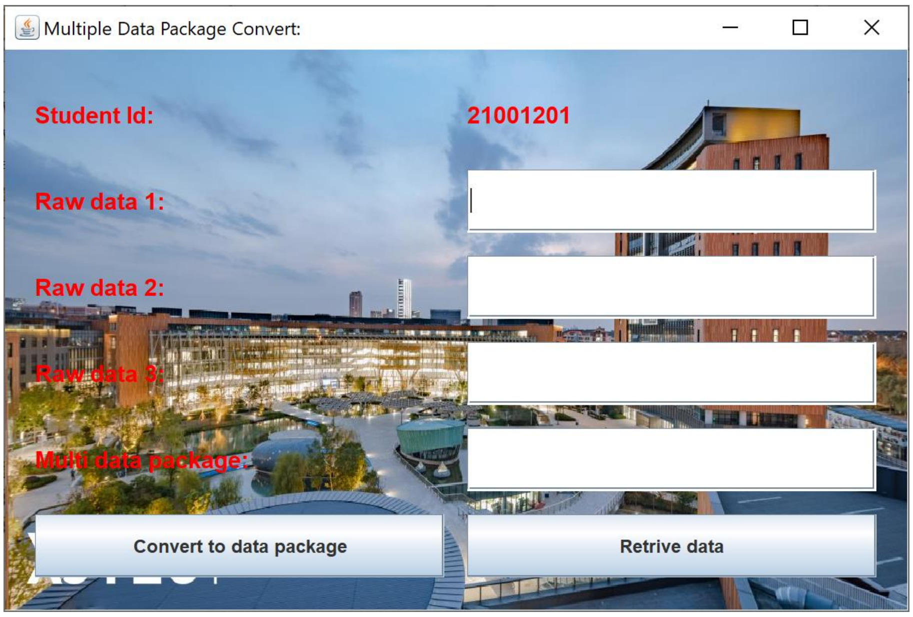
(3) Create a class "ImgePanel", which is inherited from class "JPanel".

a. Define a public constructor to create an ImagePanel object with an image file name.

b. Override the method "paintComponent (Graphics g)" to set a background image for the ImagePanel object.

(4) Create a class "Q2" and achieve the following functions in its main method.

(a) Create a frame window as below;



You should list your student Id on the top of the window, set a background image for the frame window, and set labels' font size to 15 and font color to red. You can use any image you like.

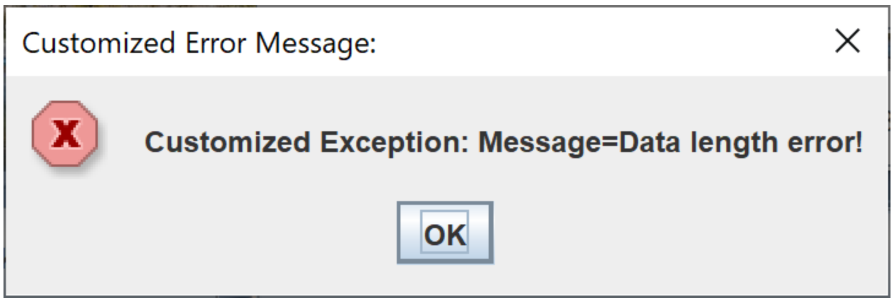
(b) Create a MultiDataPacket object in the main method;

(c) Create an ImagePanel object with a background image and add it to the Frame window as the main container of other components.

(d) Users can input strings into the first 3 text fields (raw data 1~raw data 3). When users click the "Convert to multi packet" button, strings in the 3 text fields can be encrypted and encapsulated into a hexadecimal string and shown in the "multi data packet" text field.

(e) When "Retrive data" button is clicked, users can retrieve all the separate data from the "multi data packet" text field and show them in the first 3 text fields (Raw data 1~Raw data 3), respectively.

(f) You should use a message dialogue to show any exceptions raised in your program. Below is an example for your reference.



Customized Error Message:

```
Customized Exception: Message=Data length error!
```

Resit Coursework – Part B

In this part, you will develop a car rental management system to manage information of cars, customers, staff and rental contracts.
CW3 Part B will contribute 50% of the total marks.

The system has several classes as follows:

(1) create a class “Car” to represent the car information of the car rental management system. It should have at least 3 member variables of car Id, brand name and the number of seats.

(2) Create a class “Truck” inherited from the class “Car”. It has at least 1 new member variable of “loadCapacity” to represent the load capacity of the truck.

(3) Create a class “Customer” to represent customer information. It has at least 3 member variables of customer Id, customer name and gender. Gender should be an enum type with two values of “MALE” and “FEMALE”.

(4) Create a class “Manager” to represent the manager of the car rental management system. It has at least 5 member variables of manager Id, name, gender, login name and password.

(5) Create a class “RentalInfo” to represent the rental information for customers. It has at least 3 member variables of car Id, customer Id and rental days to represent a rental record for a customer.

(6) Copy the class “Encryption” from Part A, which can be used to encrypt and decrypt the manager’s password.

(7) Create a class “FileUtils”, which can be used to create plain text files to save objects of customers, managers, cars, trucks and rental records, respectively. In addition, it should provide methods to retrieve objects from these plain text files. To achieve this, classes of customer, manager, car, truck and rental information should implement an interface of Serializable.

(8) Create a class “DataAccess”, which invokes FileUtils class to create and save objects (customers, managers, cars, trucks and rental information) into 5 different files. The class should use ArrayList to save objects retrieved from files. In addition, it can search and return objects according to the object’s id. For example, the method “Car getCarById(String carId)” can search and return a car object by the car Id.

(9) Create a class “Q3” and achieve the following functions in its main method.

    a. Create an operation menu as follows:

```
***** Operation Menu ******
1: add a customer.
2: add a car.
3: add a truck
4: add a manager.
5: add a car rental information.
6: add a truck rental information.
7: list all customers.
8: list all cars.
9: list all trucks
10: list all managers.
11: list all car rental information.
12: list all truck rental information.
13: search a customer.
14: search a car.
15: search a truck
16: search a manager.
17: search a car rental information.
18: search a truck rental information.
0: exit the program.
Please input a number to run the program:
```

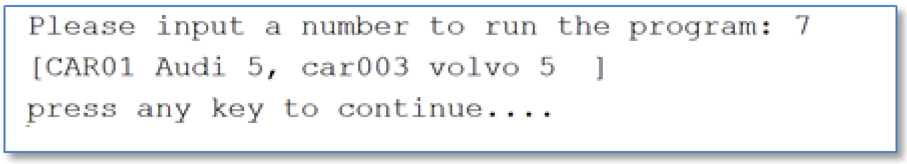
b. Users can input a number to run a related function.

c. When users input 1, then users can input customer's Id, name and gender to create a customer object and append it into a customer data file. Users can continuously create several customers until "-1" is inputted. The following picture shows that the user creates 2 customer objects, which are also saved into the customer data file.

```
Please input a number to run the program: 1
Please input customer information (input '-1' to end):
input customer Id:cus001
Input customer name:Jack
input customer gender (M/F):M
Please input customer information (input '-1' to end):
input customer Id:cus002
Input customer name:Marry
input customer gender (M/F):F
Please input customer information (input '-1' to end):
input customer Id:-1
press any key to continue....
```

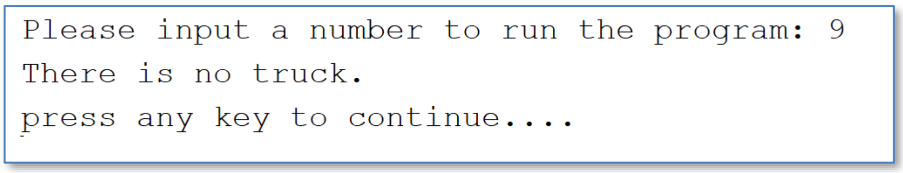
Operations to add a car, truck, manager and rental information are similar to that of adding a customer.

d. When users input 7, all the customer objects should be listed and printed out as follows.



```
Please input a number to run the program: 7
[CAR01 Audi 5, car003 volvo 5  ]
press any key to continue....
```

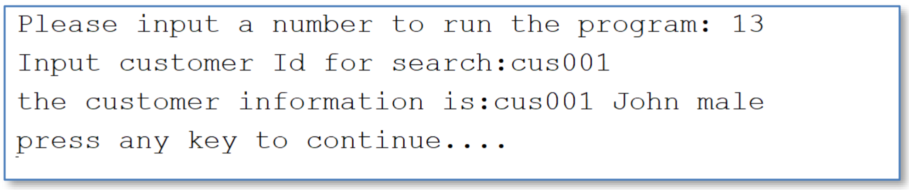
When there is no object, it should print out a message as follows to let users know that there is no object.



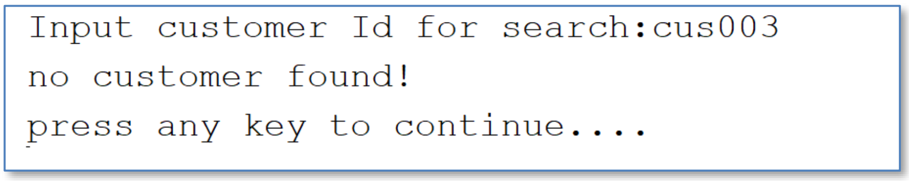
```
Please input a number to run the program: 9
There is no truck.
press any key to continue....
```

Operations to list car, truck, manager, and rental information objects are similar to that of listing student objects.

e. When users input 13, then users can input the customer Id to search and print out the related customer object. If the customer object does not exist, then print out a string to show that the customer object is not found.



```
Please input a number to run the program: 13
Input customer Id for search:cus001
the customer information is:cus001 John male
press any key to continue....
```



```
Input customer Id for search:cus003
no customer found!
press any key to continue....
```

Operations to search a car, truck, manager and rental information object are similar to that of searching a customer object. However, users should input both customer Id and car Id to search for a car or truck rental object.

(10) There are other requirements as follows:

a. The program can be continuously run until users input 0 to exit.
b. You should check and avoid duplicate Ids when you create objects.
c. Exceptions should be caught and handled properly.
d. Comments for methods and classes are required.

Resit Coursework – Part C

In Part C, you will complete the design and the documentation of the programs in Part B.
CW3 Part C will contribute to 15% of the total marks.

Javadoc Comment

Write appropriate Javadoc comments in class “RentalInfo”, generate the Javadoc HTML from that and convert and submit the Javadoc in PDF format. (5 marks)

Class Diagram

Complete the UML class diagrams of class Car, Truck and FileUtils in one PDF file. (10 marks)

* The javadoc and class diagrams should be put into one single PDF report for submission.

Resit Coursework – Additional Notes

Submission instructions

You should follow the submission steps below to submit your documents to the Learning Mall.

(1) You should complete and submit the resit coursework before July 28, 2022.
(2) Zip the Whole NetBeans project folder of Part A into one zipped file and upload it to the learning mall using the Part A submission link.
(3) Zip the whole NetBeans project folder of Part B into one zipped file and upload it to the learning mall using the Part B submission link.
(4) Zip all the documents of Part C into one zipped file and upload it to the learning mall using the Part C submission link.

Late submissions: The standard University policy on late submissions will apply: 5% of the total marks available for the component shall be deducted from the assessment mark for each working day after the submission date, up to a maximum of five working days, so long as this does not reduce the mark below the pass mark (40%). Submissions more than five working days late will not be accepted.

On Plagiarism

This resit coursework is individual work. Plagiarism (e.g. copying materials from other sources without proper acknowledgement) is a serious academic offence. Plagiarism and collusion will not be tolerated and will be dealt with in accordance with the University Code of Practice on Academic Integrity. Individual students may be invited to explain parts of their code in person, and if they fail to demonstrate an understanding of the code, no credit will be given for that part of the code.

This is the end of resit Coursework Task Sheet.