1111 物件導向設計與實習(資訊二合)[3670/3671]期末考試

考試說明

✓ 考試時間：2023 年 1 月 3 日 18:30~21:30

- (最早交卷時間 20:30，最後交卷時間 21:30)。

✓ 考試地點：資電 234 電腦教室，必須到現場簽到並簽署考試公約。

- (有特殊狀況同學請聯繫老師)

✓ 考試題目：公告於 iLearn 2.0 期中考試題目區，可先行思考解題方式。

✓ 作答方式：

- 每題的答案至少須包含程式碼與執行結果，有簡要解題方法說明為佳。請將個人最後作答資料轉成「Fexam_學號.pdf」檔案於最後交卷時間前交到 iLearn 2.0 期中考試答案區。
- 考試題目中若有提供程式碼樣板(skeleton.zip)，請盡量以程式碼樣板為基礎作答，但不限於只可用該程式碼樣板作答。**原始作答的使用到的相關程式碼(含註解)、程式輸出入資料畫面等，可以於答案紙中清楚註明參考附件檔名或逕行貼於答案紙中，一併交到 iLearn 2.0 期中考試答案區。**

✓ 計分原則：

- 考試題目每題會標註配分百分比[X%]，視作答正確性與題目要求完整性給予 X 內的分數。
- 對於特定程式撰寫題目之作答超過題目基本要求並具備作法描述者，該配分得增加 [X%+Y%]中之 Y%加碼，該題以 X+Y 內的分數評分。
- 本次配分含加碼總計 120 分，總分超過 100 分者最高以 100 分計算。

✓ 注意事項：

- 本次期中考試可利用書籍或使用電腦上網找尋相關資料，但**禁止於考試過程中將先行公告題目之答案以各種具體形式帶入考試地點**。
- 考試期間請同學將手機關機，禁止同學相互交談，或使用隨身碟、各類通訊設備、線上社群與同學討論。
- **部分違反上述規則者，以作答總分*60%計算；嚴重抄襲者，以該題作答總分*20%計算。**

敬祝大家考試順利　作答愉快!!!

考試題目

PART I: Brief Answer Each of the Following Questions [30%]

1. What are three major **characteristics of object-oriented** programming? Explain their benefits. **[12%]**
2. What is a **Java interface**? Explain how a member function in a Java interface is defined and implemented. **[8%]**
3. What are the five keywords used in **Java Exception**? Explain their usages. **[10%]**

(Continue in the next page)

PART II. Programming Questions [70%+20%]

1. Currency Exchange [18%] (計分標準：依正確性與完整度給分 4%~18%)

The **abstract class** Currency specifies money currency. The abstract class is **extended** to three concrete sub-classes **class** NTD for New Taiwan Dollar, **class** USD for US Dollar, and **class** JPY for Japanese Yen.

Assume the exchange rate of 1 US Dollar is equal to 30.78 New Taiwan Dollar and 133.52 Japanese Yen;

the exchange rate of 1 New Taiwan Dollar is equal to 0.032 US Dollar and 4.34 Japanese Yen;

the exchange rate of 1 Japanese Yen is equal to 0.0075 US Dollar and 0.23 New Taiwan Dollar.

Complete the following steps:

(a) Complete the implementation of the constructors in **class** NTD.

(b) Complete the implementation of the constructors in **class** USD.

(c) Complete the implementation of the constructors in **class** JPY.

(d) Complete the implementation of function printc() to print the currency amount in **class** NTD, **class** USD, and **class** JPY.

(e) The foreign exchange policy in Taiwan requires that any foreign currency exchange buying or selling must be via New Taiwan Dollar. Suppose a person is going to buy US Dollar using 300,000 Japanese Yen.

Write a Java main function to compute his/her exchange gain/loss, if he/she could directly buy US Dollar using Japanese Yen.

(Continue in the next page)

2. Tic-tac-toe [20%+12%]

(計分標準：依答案完整度與正確性給分 4%~20%，針對使用者操作介面改用 **GUI Layout 互動者**最多加碼 12%)

Tic-tac-toe (Xs and Os) is a paper-and-pencil game for two players who take turns marking the spaces in a three-by-three grid cells with X or O. The player who succeeds in placing three of their marks in a row, column, diagonal, or anti-diagonal is the winner.

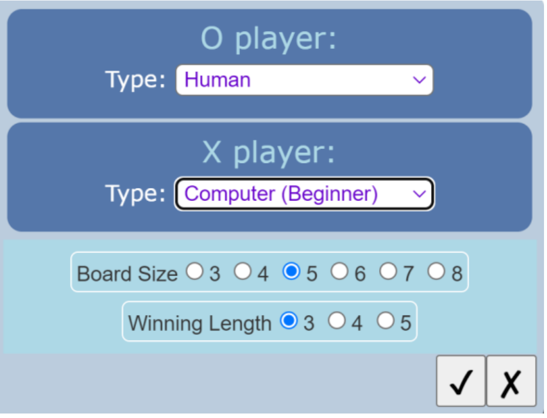
Write a Java console or GUI (such as JavaFX) program to implement the tic-tac-toe game with a user player and computer. Different board sizes such as 3*3, 4*4, 5*5, …, 8*8 can be set by the player. The winning length can be set 3, 4, 5 as well.

In the following examples,

- Console mode: the boarder size is 3*3, the winning length is 3. You can implement it by interactive console mode. The player wins the game in seven steps:

  Player: 0,2 / Computer: 0,0 / Player: 2,1 / Computer 1,1 / Player: 2,2 / Computer: 1,2 / Player:2,1
  ➔ Game Over. The Player Wins!!!

- GUI mode: the boarder size is 5*5, the winning length is 3. You can implement it by interactive by GUI mode. The player (O) wins the game. *(source: https://www.mathsisfun.com/games/tic-tac-toe.html)*

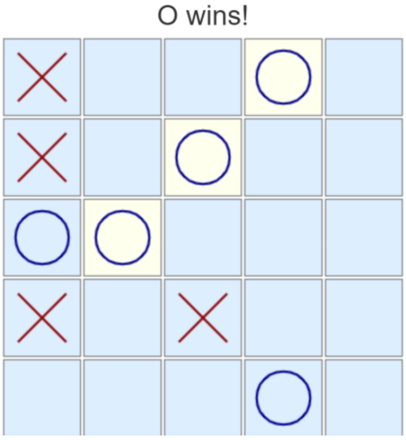


O player:
Type: Human

X player:
Type: Computer (Beginner)

Board Size ○3 ○4 ●5 ○6 ○7 ○8

Winning Length ●3 ○4 ○5



O wins!

X			O	
X		O		
O	O			
X		X		
			O	

(Continue in the next page)

3. Image File Processing [32%+8%]

(計分標準：依正確性與完整度給分 6%~32%，**程式中加上 Exception Handling** 者最多加碼 8%)

In image processing, a bitmap is a type of file format to store digital images. The format of a bitmap file is divided into two parts, *file header and image information*, defined as follows *and refer to BitmapImage.java in skeleton*.

```
// File header: 14 bytes.
byte[2] type; // Two fixed characters, "BM" for bitmap images.
int size; // File size in bytes.
byte[4] reserved; // Reserved field.
int offsetBits; // Offset, starting address where the bitmap image data are stored.
// Image information: 40 bytes.
int infoSize; // Information size in byte.
int width; // Image width in pixel.
int height; // Image height in pixel.
short planes; // Number of image planes in the image, must be 1.
short bitPerPixel; // Number of bits used to represent the data for each pixel.
int compression; // Compressed form, 0 is uncompressed.
int imageSize; // Size of the actual pixel data, in bytes.
int xResolution; // Horizontal resolution of the image, pixels per meter.
int yResolution; // Vertical resolution of the image, pixels per meter.
int colors; // Color indices, value is 0.
int importantColors;   // Number of important colors, value is 0.
```

Following the file header and image information, there is the color palette with the size of OffsetBits-InfoSize-14 bytes, and then the image pixel data with the size of ImageSize bytes. The color palette and image pixel data can be declared as pointers of unsigned char. The images used in this programming practice does not use color palette, i.e., OffsetBits-InfoSize-14 is 0.

The pixel data area is stored using the following format:

- ✓ Image pixels are stored in the row-major order starting from the bottom-left corner. That is, pixels are stored from left to right and from bottom to top; the pixels in the last row are stored on the front of the pixel data area.
- ✓ Each pixel takes three bytes representing levels of the three original colors red, green, and blue (RGB) one byte each color. The order of the three bytes is blue color in the first byte, green in the second byte and red in the third byte.
- ✓ The number of bytes in a row must be a multiple of four. If the number of actual pixel bytes is not a multiple of four, 0X00's are padded at the end of the row.

3.1 Image Copy [20%+6%]

Use ***class*** *BitmapImage in* ***package*** *or define BitmapImage is a* ***static inner class*** to write a Java program and to perform the following steps:

(1) read in a bitmap image, *(using exception mechanism to handle the situation if the input file is not existed)*

(2) copy the image into another bitmap image,

(3) ***print the header and image information of the original and copied bitmap.***

```
Enter the source image file name: abraham_lake.bmp
Enter the banded filter image file name: copied.bmp
==========================================================
Image file name: C:\ImageApplications\ abraham_lake.bmp
type: BM
size: 1080054
reserved:
offsetBits: 54
information size: 40
image width: 800
image height: 450
image planes: 1
bit per pixel: 24
compression: 0
image size: 1080000
horizontal resolution: 5669
vertical resolution: 5669
color indices: 0
important colors: 0
==========================================================
Image file name: C:\ImageApplications\copied.bmp
type: BM
size: 1080054
reserved:
offsetBits: 54
information size: 40
image width: 800
image height: 450
image planes: 1
bit per pixel: 24
compression: 0
image size: 1080000
horizontal resolution: 5669
vertical resolution: 5669
color indices: 0
important colors: 0
```

3.2 Image Flip and Merge [12%+2%]

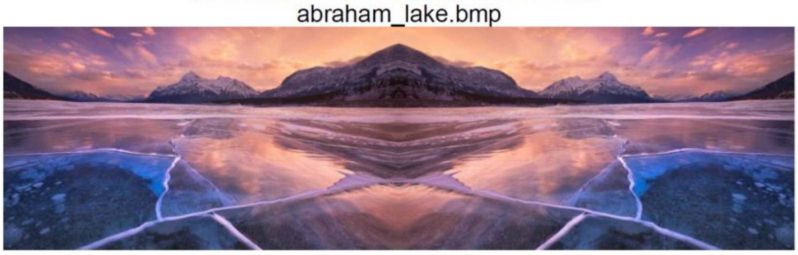
*Use class BitmapImage in **package** or define BitmapImage is a **static inner class*** to write a Java program to perform flip and merge of a bitmap image. The flip and merge operation is to perform horizontal mirror reflection of the bitmap image and then paste it on the right hand side of the original image. Write a Java program to perform the following steps:

(1) read in a bitmap image, *(using exception mechanism to handle the situation if the input file is not existed)*

(2) flip the bitmap image horizontally along the vertical center line, and then merge it on the right side of the original image,

(3) ***output the merged bitmap image*.**

abraham_lake.bmp



flip_merge.bmp

You may use example color image file (abraham_lake.bmp) or other image files to test your program. If you use a color image other than the bit map format, such as png, gif, or jpg format, you may use a graphics tool to convert it to a bmp format file.

-- The End --