

類 科：資訊處理

科 目：程式語言

考試時間：2 小時

座號：_____

※注意：(一)禁止使用電子計算器。

(二)不必抄題，作答時請將試題題號及答案依照順序寫在試卷上，於本試題上作答者，不予計分。

(三)本科目除專門名詞或數理公式外，應使用本國文字作答。

- 一、閱讀以下 Java 程式，列出數學式以說明變數 e 在計算什麼？接著撰寫遞迴 (recursive) 程式 `public static double etx (int accuracy, int x)` 來計算前述變數 e 的值，撰寫時必須使用 `etx` 規定的參數與資料型態。(25 分)

```
import java.util.Scanner;

public class EtoX
{
    public static void main( String[] args )
    {
        Scanner input = new Scanner( System.in );
        int number = 1;
        int accuracy;
        int factorial = 1;
        int x;
        double e = 1.0;
        double exponent = 1.0;

        x = input.nextInt();
        accuracy = input.nextInt();

        while ( number < accuracy )
        {
            exponent *= x;
            factorial *= number;
            e += exponent / factorial;
            number++;
        } // end while loop

        System.out.printf( "x: %d%ne: %f%n", x, e );
    } // end main
} // end class EtoX
```

二、下列為資料結構 List 的 Java 程式，而 ListTest 為測試類別（class），試回答以下問題：（35 分）

- (1) ListTest 中 "List<Integer> list = new List<>();" 會先後呼叫那些 methods？傳送那些參數值？結果新物件 list 的屬性值為何？
- (2) 執行 ListTest.java 後會列印出什麼？
- (3) 撰寫 public T removeFromBack() throws EmptyListException。

```
class ListNode<T>
{
    T data;
    ListNode<T> nextNode;

    ListNode(T object)
    {
        this(object, null);
    }
    ListNode(T object, ListNode<T> node)
    {
        data = object;
        nextNode = node;
    }
    T getData()
    ListNode<T> getNext()
} // end class ListNode<T>
```

```
public class List<T>
{
    private ListNode<T> firstNode;
    private ListNode<T> lastNode;
    private String name;

    public List()
    {
        this("list");
    }
    public List(String listName)
    {
```

```
        name = listName;
        firstNode = lastNode = null;
    }

    public void insertAtFront(T insertItem)

    public void insertAtBack(T insertItem)

    public T removeFromFront() throws EmptyListException

    public T removeFromBack() throws EmptyListException

    public boolean isEmpty()

    public void print()
    {
        if (isEmpty())
        {
            System.out.printf("Empty %s%n", name);
            return;
        }
        System.out.printf("The %s is: ", name);
        ListNode<T> current = firstNode;
        while (current != null)
        {
            System.out.printf("%s ", current.data);
            current = current.nextNode;
        }
        System.out.println();
    }
} // end class List<T>

public class EmptyListException extends RuntimeException
{
    public EmptyListException()
    {
        this("List");
    }
}
```

```
public EmptyListException(String name)
{
    super(name + " is empty");
}
} // end class EmptyListException

public class ListTest
{
    public static void main(String[] args)
    {
        List<Integer> list = new List<>();

        try
        {
            list.insertAtFront(-1);
            list.insertAtFront(99);
            list.print();
            int removedItem = list.removeFromFront();
            removedItem = list.removeFromFront();
            list.print();
            removedItem = list.removeFromFront();
            list.print();
        }
        catch (EmptyListException emptyListException)
        {
            emptyListException.printStackTrace();
        }
    }
} // end class ListTest
```

三、下列程式 Stack<T> 繼承上題的 List<T>。試撰寫 Stack 中的建構子 Stack()，以及兩個主要 methods: push(...)與 pop()。(25 分)

```
public class Stack<T> extends List<T>
{
    public Stack()

    public void push(T object)

    public T pop() throws EmptyListException
} // end class StackInheritance
```

四、下列 Python 程式的執行結果為何？(15 分)

```
class Shape:
    def __init__(self, x, y):
        self.x = x
        self.y = y
        self.description = "unknown"
    def area(self):
        return self.x * self.y
    def perimeter(self):
        return 2 * self.x + 2 * self.y
    def describe(self, text):
        self.description = text
```

```
class Square(Shape):
    def __init__(self, x):
        self.x = x
        self.y = x
```

```
class DoubleSquare(Square):
    def __init__(self, y):
        self.x = 2 * y
        self.y = y
    def perimeter(self):
        return 2 * self.x + 3 * self.y
```

```
rectangle = Shape(100, 45)  
print(rectangle.perimeter())
```

```
dictionary = {}  
dictionary["DoubleSquare"] = DoubleSquare(5)  
dictionary["Rectangle"] = Shape(600,45)  
dictionary["Square"] = Square(20)  
print(dictionary["Square"].area())
```

```
dictionary["DoubleSquare"].describe("Double square")  
print(dictionary["Rectangle"].description)
```