Chú ý:

Tạo 1 Project đặt tên là "TestingSystem_Assignment_6"

Tạo 3 package entity, backend, front-end

Trong phần backend sẽ tạo các class Exercise1, Exercise2, … mỗi method là 1 Question

Trong front-end tạo các class Program1, Program2, Program3, … để demo kết quả của các Excercise1, Excercise2, Excercise3, …

Exercise 1: Debug

Question 1: 1 học viên đã code bài min,max và có 1 vài lỗi sai như hình dưới (Code trong File DebugExercise.rar, hãy giải nén ra và import vào eclipse). Để tìm được ra bug, hãy thực hiện từng thao tác dưới đây để tìm ra lỗi và sửa nó

a) Trong function getMaxValue, hãy tìm xem tại vòng lặp i = 4 thì variable maxValue đang có giá trị bao nhiêu

b) Tại vòng lặp i = 4, hãy thử set lại maxValue = 5

c) Hãy tìm bug và sửa lại cho bài trên để tìm Max Value và Min Value cho đúng

```
3  public class Question1 {
4      public static void main(String[] args) {
5
6          int[] numbers = { -1, 1, 3, 5, -5, -8 };
7          int maxValue = getMaxValue(numbers);
8          int minValue = getMinValue(numbers);
9
10         System.out.println("Max Value: " + maxValue);
11         System.out.println("Min Value: " + minValue);
12     }
13
14     public static int getMaxValue(int[] numbers) {
15         int maxValue = numbers[0];
16         for (int i = 1; i < numbers.length; i++) {
17             if (numbers[i] < maxValue) {
18                 maxValue = numbers[i];
19             }
20         }
21         return maxValue;
22     }
23
24     public static int getMinValue(int[] numbers) {
25         int minValue = numbers[0];
26         for (int i = 1; i < numbers.length - 1; i++) {
27             if (numbers[i] < minValue) {
28                 minValue = numbers[i];
29             }
30         }
31         return minValue;
32     }
33 }
```

```
Console  Pro
<terminated> Exerci
Max Value: -8
Min Value: -5
```

Question 2: Trong bài tập Assignment 4 (File Testing_System_Assignment_4_Debug.rar, giải nén ra và import vào eclipse), có 1 học viên đã làm như trong project, nhưng khi chạy chương trình học viên đó phát hiện chương trình đã chạy sai, cụ thể như sau:

B1: Học viên chọn nhập chức năng thêm mới cán bộ

B2: Chọn thêm 1 cán bộ vào

B3: nhập thông tin cán bộ vào

Nhưng học viên đã phát hiện ra hệ thống vẫn yêu cầu nhập thêm người nữa, hãy debug để tìm ra lỗi và sửa lỗi hộ bạn.

```
Console  Problems  Javadoc  Declaration  Search  Progress
Program (25) [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe  (Oct 8, 2020, 9:34:23 AM)
Mời bạn nhập vào chức năng muốn dùng
1.Thêm mới cán bộ
2.Tìm kiếm theo họ tên
3.Hiện thị thông tin về danh sách các cán bộ.
4.Nhập vào tên của cán bộ và delete cán bộ đó
5.Thoát khỏi chương trình.
Mời bạn chọn chức năng: 1                                   B1

Bạn muốn add bao nhiêu cán bộ: 1                            B2

Bạn muốn nhập vào nhân viên(nhập vào 1), Công nhân (nhập vào 2), kỹ sư (nhập vào 3): 2
Mời bạn nhập vào họ tên: Nguyen Van A                       B3
Mời bạn nhập vào tuổi: 20
Mời bạn nhập vào Bac: 3
Nhập thành Cán bộ 1

Bạn muốn add bao nhiêu cán bộ:
```

error

Exercise 2: Exception

Question 1: try…catch…finally

```
3  public class Program2 {
4      public static void main(String[] args) {
5          float result = divide(7, 0);
6
7          System.out.println(result);
8      }
9
10     public static float divide(int a, int b) {
11         return a / b;
12     }
13 }
```

Hãy xử lý exception cho VD trên, khi bị lỗi thì sẽ in ra text "cannot divide 0"

Tạo Class Exercise2

```
package com.vti.backend;

public class Exercise2 {
	public static Float device(int a, int b) {
		return (float) (a / b);
	}
}
```

Tạo Class Demo_Exercise2

```
package com.vti.frontend;

import java.util.Scanner;

import com.vti.backend.Exercise2;

public class Demo_Exercise2 {

	public static void main(String[] args) {
		question1();
	}

	private static void question1() {
		Scanner sc = new Scanner(System.in);
		System.out.println("Nhập vào số 1: ");
		int num1 = sc.nextInt();
		System.out.println("nhập vào số 2: ");
		int num2 = sc.nextInt();
		try {
			System.out.println("Kết quả phép chia 2 số là: " +
Exercise2.device(num1, num2));
		} catch (Exception e) {
			System.err.print("có lỗi khi thực hiện chia cho số 0");
		}

	}
}
```

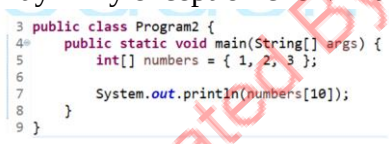

Question 2: tiếp tục Question 1
Sau khi thực hiện xong method divide dù có lỗi hay không sẽ in ra text "divide completed!" ➔ thêm block **finally**

```
package com.vti.frontend;

import java.util.Scanner;

import com.vti.backend.Exercise2;

public class Demo_Exercise2 {

	public static void main(String[] args) {
		question1();
	}

	private static void question1() {
```

```
		Scanner sc = new Scanner(System.in);
		System.out.println("Nhập vào số 1: ");
		int num1 = sc.nextInt();
		System.out.println("nhập vào số 2: ");
		int num2 = sc.nextInt();
		try {
			System.out.println("Kết quả phép chia 2 số là: " +
Exercise2.device(num1, num2));
		} catch (Exception e) {
			System.err.print("có lỗi khi thực hiện chia cho số 0");
		}finally {
			System.out.println("");
			System.out.println("Hoàn thành");
		}

	}
}
```

Question 3:

Hãy xử lý exception cho VD trên và in ra thông tin lỗi.



```
public class Program2 {
    public static void main(String[] args) {
        int[] numbers = { 1, 2, 3 };

        System.out.println(numbers[10]);
    }
}
```

```
private static void question3() {
		int[] ints = { 1, 4, 6 };
		try {
			System.out.println(ints[10]);
		} catch (Exception e) {
			System.err.print("Không tìm thấy phần tử mảng");
		}
	}
```

Question 4:

Tạo 1 array departments gồm 3 phần tử
Sau đó viết 1 method getIndex(int index) để lấy thông tin phần tử thứ index trong array departments. Nếu index vượt quá length lấy ra thì sẽ in ra text "Cannot find department."

➔Tạo class department

```
package com.vti.entity;

public class Department {
	public static int COUNT;
	private int id;
	private String name;

	public Department(String name) {
		super();
		COUNT++;
		this.id = COUNT;
		this.name = name;
	}
```

```
	@Override
	public String toString() {
		return "Department [id=" + id + ", name=" + name + "]";
	}

}
```

➔Tạo class để demo

```
private static void question4() {
		Department dep1 = new Department("Dep1");
		Department dep2 = new Department("Dep2");
		Department dep3 = new Department("Dep3");
		Department dep4 = new Department("Dep4");
		Department[] depArray = { dep1, dep2, dep3, dep4 };

		try {
			System.out.println(depArray[10]);
		} catch (Exception e) {
			System.err.println("Không tìm thấy phần tử này trong danh
sách.");
		}
	}
```

Question 5:

➔ Hướng dẫn sinh viên tạo 1 class ScannerUltis để sử dụng sau này:

Tạo 1 method inputAge() và trả về 1 số int.
Trong method hãy cài đặt như sau:
B1: Sau đó dùng scanner để nhập vào 1 số
B2: Check exeption
Nếu người dùng nhập vào 1 số thì return về số đó
Nếu người dùng không nhập vào 1 số thì sẽ in ra dòng text "wrong inputing! Please input an age as int, input again."
Nếu người dùng không nhập vào 1 số < 0 thì sẽ in ra dòng text "Wrong inputing! The age must be greater than 0, please input again."

➔Tạo class ScannerUtils

```
package com.vti.ultis;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.util.Scanner;
import java.util.regex.Pattern;

public class ScannerUltis {
	private static Scanner sc = new Scanner(System.in);

	public static int inputInt() {
		while (true) {
			try {
```

```
                        return Integer.parseInt(sc.next().trim());
                } catch (Exception e) {
                        System.err.println("Nhập lại:");
                }
        }
}

public static int inputIntPositive() {
        while (true) {
                try {
                        int intPositive = Integer.parseInt(sc.next());
                        if (intPositive >= 0) {
                                return intPositive;
                        } else {
                                System.err.println("Nhập lại:");
                        }

                } catch (Exception e) {
                        System.err.println("Nhập lại:");
                }

        }
}

public static Float inputFloat(String mes) {
        while (true) {
                try {
                        return Float.parseFloat(sc.next());
                } catch (Exception e) {
                        System.err.println("Nhập lại:");
                }
        }
}

public static Double inputDouble() {
        while (true) {
                try {
                        return Double.parseDouble(sc.next());
                } catch (Exception e) {
                        System.err.println("Nhập lại:");
                }
        }
}

public static String inputString() {
        while (true) {
                String string = sc.nextLine().trim();
                if (!string.isEmpty()) {
                        return string;
                } else {
                        System.err.println("Nhập lại:");
                }
        }
}

public static LocalDate inputLocalDate() {
        System.out.println("Nhập theo định dạng yyyy-MM-dd");
        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
        while (true) {
```

```
                        String localdate = sc.next().trim();
                        try {
                                if (format.parse(localdate) != null) {
                                        LocalDate lc = LocalDate.parse(localdate);
                                        return lc;
                                }
                        } catch (Exception e) {
                                System.err.println("Nhập lại:");
                        }

                }
        }
}
```

B3: hãy demo trong method main()

➔Tạo class Question56 chứa main để demo chương trình

```
        private static void question5() {
                System.out.println("Nhập vào tuổi của sinh viên: ");
                int age = ScannerUltis.inputIntPositive();
                System.out.println("Tuổi của sinh viên là: " + age);
        }
```

Gợi ý:

- Tại bước 1 & bước 2: Check exception, ta có thể yêu cầu người dùng nhập vào String (scanner.nextLine()),
- Sau đó sử dụng casting datatype để convert String to int
- Nếu convert được thì suy ra người dùng nhập vào 1 số int
- Nếu không convert được thì suy ra người dùng nhập sai, khi người dùng nhập sai thì sẽ in ra text "wrong inputing! Please input an age as int, input again."

=>Sử dụng hàm scanner.nextLine() để tránh bị lặp khi sử dụng scanner.nextint,…

Question 6: Tiếp tục Question 5

Sửa lại method inputAge() như sau:

Tại B2 của Question 5, Nếu người dùng không nhập vào 1 số thì sẽ in ra dòng text "wrong inputing! Please input an age as int, input again.", đồng thời yêu cầu người dùng nhập lại

```
public static int inputIntPositive() {
                while (true) {
                        try {
                                int intPositive = Integer.parseInt(sc.next());
                                if (intPositive >= 0) {
                                        return intPositive;
                                }else {
                                        System.err.println("Nhập lại:");
                                }

                        } catch (Exception e) {
                                System.err.println("Nhập lại:");
```

```
                }

            }
        }
```

Gợi ý: sử dụng while

Question 7: làm tương tự câu 6
Tạo 1 class ScannerUtils, trong class sẽ tạo 1 method inputInt() chuyên để nhập dữ liệu dạng int như age, id, …
Gợi ý: inputInt() sẽ có parameter là String errorMessage để người dùng có thể tự điền được errorMessage vào
➔Class ScannerUtils

```
package com.vti.ultis;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.util.Scanner;
import java.util.regex.Pattern;

public class ScannerUltis {
	private static Scanner sc = new Scanner(System.in);

	public static int inputInt() {
		while (true) {
			try {
				return Integer.parseInt(sc.next().trim());
			} catch (Exception e) {
				System.err.println("Nhập lại:");
			}
		}
	}

	public static int inputIntPositive() {
		while (true) {
			try {
				int intPositive = Integer.parseInt(sc.next());
				if (intPositive >= 0) {
					return intPositive;
				} else {
					System.err.println("Nhập lại:");
				}

			} catch (Exception e) {
				System.err.println("Nhập lại:");
			}

		}
	}

	public static Float inputFloat(String mes) {
		while (true) {
			try {
				return Float.parseFloat(sc.next());
			} catch (Exception e) {
```

```
                        System.err.println("Nhập lại:");
                    }
                }
            }

            public static Double inputDouble() {
                while (true) {
                    try {
                        return Double.parseDouble(sc.next());
                    } catch (Exception e) {
                        System.err.println("Nhập lại:");
                    }
                }
            }

            public static String inputString() {
                while (true) {
                    String string = sc.nextLine().trim();
                    if (!string.isEmpty()) {
                        return string;
                    } else {
                        System.err.println("Nhập lại:");
                    }
                }
            }

            public static LocalDate inputLocalDate() {
                System.out.println("Nhập theo định dạng yyyy-MM-dd");
                SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
                while (true) {
                    String localdate = sc.next().trim();
                    try {
                        if (format.parse(localdate) != null) {
                            LocalDate lc = LocalDate.parse(localdate);
                            return lc;
                        }
                    } catch (Exception e) {
                        System.err.println("Nhập lại:");
                    }

                }
            }
        }
```


Question 8: làm tương tự câu 7

Làm tương tự câu 7 với các method inputFloat(), inputDouble(), inputString()

Riêng với inputString() thì không cần phải handling exception

Đáp án trên câu 7

Question 9:
Sử dụng ScannerUtils vừa tạo để nhập thông tin cho Department, Position với điều kiện khi khởi tạo object sẽ yêu cầu nhập vào thông tin luôn trong constructor (VD như hình dưới)

```
public class Student {

    private Scanner scanner;

    private String name;
    private int age;

    public Student() throws Exception {
        scanner = new Scanner(System.in);

        name = inputName();
        age = inputAge();
    }

    private int inputAge() throws Exception {
        try {

            System.out.print("Please input your age: ");
            int age = scanner.nextInt();

            return age;

        } catch (Exception e) {
            throw new Exception("Please input a number as int");
        }
    }

    private String inputName() throws Exception {
        try {

            System.out.print("Please input your name: ");
            String name = scanner.next();

            return name;

        } catch (Exception e) {
            throw new Exception("Please input a name");
        }
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
```

➔ Tạo class `Position`

```
package com.vti.entity;

import com.vti.ultis.ScannerUltis;

public class Position {
	public static int COUNTPos = 0;
	public int id;
	public PositionName name;

	public enum PositionName {
		Dev, Test, Scrum_Master, PM

	}

	@Override
	public String toString() {
		return "Position [id=" + id + ", name=" + name + "]";
	}

	public Position() {
		super();
		COUNTPos++;
		this.id = COUNTPos;
		System.out.println("Chọn tên vị trí muốn nhập 1.Dev, 2.Test,
3.Scrum_Master, 4.PM : ");

		while (true) {
			int choosePos = ScannerUltis.inputIntPositive();
			switch (choosePos) {
			case 1:
```

```
                        this.name = PositionName.Dev;
                        return;

                    case 2:
                        this.name = PositionName.Test;
                        return;

                    case 3:
                        this.name = PositionName.Scrum_Master;
                        return;

                    case 4:
                        this.name = PositionName.PM;
                        return;
                        default:
                            System.err.println("Chọn lại: ");
                    }
                }
        }

}
```

➔Tạo class Department

```
package com.vti.entity;

import com.vti.ultis.ScannerUltis;

public class Department {
        public static int COUNT;
        private int id;
        private String name;

        public Department(String name) {
                super();
                COUNT++;
                this.id = COUNT;
                this.name = name;
        }

        public Department() {
                super();
                COUNT++;
                this.id = COUNT;
                System.out.println("Nhập tên phòng: ");
                this.name = ScannerUltis.inputString();
        }

        @Override
        public String toString() {
                return "Department [id=" + id + ", name=" + name + "]";
        }

}
```

➔Tạo class để demo:

```
private static void question9_Pos() {
                System.out.println("Tạo Position");
```

```
		Position pos1 = new Position();
		System.out.println("Thông tin Position vừa nhập: ");
		System.out.println(pos1);
	}
```

```
private static void question9_Dep() {
		Department dep1 = new Department();
		Department dep2 = new Department();
		Department dep3 = new Department();
		ArrayList<Department> listDep = new ArrayList<Department>();
		listDep.add(dep1);
		listDep.add(dep2);
		listDep.add(dep3);
		System.out.println("Thông tin phòng vừa nhập: ");
		for (Department department : listDep) {
			System.out.println(department);
		}

	}
```

Question 10: làm giống bài 9
Sử dụng ScannerUtils để nhập thông tin cho Group
Khi tạo các property array accounts thì hỏi người dùng xem, bạn có muốn thêm accounts hay không, nếu người dùng đồng ý thì sẽ nhập thông tin account
(với mỗi thông tin Account nhập vào ta sẽ tạo Object mới)
➔Tạo Class Account:

```
package com.vti.entity;

import java.time.LocalDate;

import com.vti.ultis.ScannerUltis;

public class Account {
	public static int COUNT = 0;
	private int id;
	private String email;
	private String userName;
	private String fullName;

	@Override
	public String toString() {
		return "Account [id=" + id + ", email=" + email + ", userName=" +
userName + ", fullName=" + fullName + "]";
	}

	public Account() {
		super();
		System.out.println("Nhập thông tin Account: ");
		COUNT++;
		this.id = COUNT;
		System.out.println("Nhập tên Email: ");
		this.email = ScannerUltis.inputString();
		System.out.println("Nhập tên userName: ");
```

```
		this.userName = ScannerUltis.inputString();
		System.out.println("Nhập tên fullName: ");
		this.fullName = ScannerUltis.inputString();
	}

}
```

➔Tạo class Group:

```
package com.vti.entity;

import java.time.LocalDate;
import java.util.Arrays;
import java.util.Iterator;

import com.vti.ultis.ScannerUltis;

public class Group {
	public static int COUNT = 0;
	private int id;
	private String name;
	private Account creator;
	private LocalDate createDate;
	private Account[] accounts;

	public void printInforGroup() {
		System.out.println("Thông tin Group:");
		System.out.println("Group [id=" + id + ", name=" + name + ",
createDate=" + createDate + "]");
		if (accounts != null) {
			System.out.println("Số lượng Account trong Group này là: " +
accounts.length);
			for (Account account : accounts) {
				System.out.println(account.toString());
			}
		}

	}

	public Group() {
		super();
		System.out.println("Nhập thông tin tạo Group: ");
		COUNT++;
		this.id = COUNT;
		System.out.println("Nhập tên Group: ");
		this.name = ScannerUltis.inputString();
		System.out.println("Nhập ngày tạo Group: ");
		this.createDate = ScannerUltis.inputLocalDate();
		System.out.println("Bạn có muốn thêm Account vào Group hay không,
1.Có, 2.Không");

		while (true) {
			int chooseAddAcc = ScannerUltis.inputIntPositive();
			switch (chooseAddAcc) {
			case 1:
				System.out.println("Nhập số lượng account muốn thêm
vào Group này: ");
```

```
                        int countAcc = ScannerUltis.inputIntPositive();
                        Account[] accs = new Account[countAcc];
                        for (int i = 0; i < countAcc; i++) {
                                System.out.println("Nhập vào Account thứ " + (i
+ 1) + ": ");

                                Account acc = new Account();
                                accs[i] = acc;
                        }
                        this.accounts = accs;
                        return;
                case 2:
                        return;

                default:
                        System.out.println("Nhập lại: ");
                        break;
                }
        }

    }

}
```

➔Tạo class để demo chương trình

```
package com.vti.backend;

import com.vti.entity.Group;

public class Exercise2_Ques10 {

    public void question10() {
        System.out.println("Tạo group: ");
        Group gp = new Group();
        gp.printInforGroup();
    }

}
```

Question 11 (Optional): Custom Exception

Tạo custom Exception tên là: InvalidAgeInputingException.
Hãy viết method inputAge() ở class Account với yêu cầu như sau:
Khi người dùng nhập vào tuổi <= 0 thì throw ra Exception InvalidAgeInputingException với message là: "The age must be greater than 0"

➔Tạo class InvalidAgeInputingException

```
package com.vti.backend;

public class InvalidAgeInputingException extends Exception {
    public InvalidAgeInputingException(String mes) {
        super(mes);
    }
}
```

Question 12 (Optional): Tiếp tục Question 11
Trong class Account. Sử dụng method inputAge() từ ScannerUtils, hãy viết method inputAccountAge() với yêu cầu như sau:
Khi hệ thống gặp exception InvalidAgeInputingException thì sẽ in ra message.
Và khi người dùng nhập vào tuổi < 18 thì in ra text "Your age must be greater than 18" và yêu cầu người dùng nhập lại tuổi
Sau đó sử dụng method inputAccountAge() để nhập tuổi ở trong Constructor (sửa lại Constructor ở Question 10)

➔ Tạo class AccountQues11 để sử dụng exception InvalidAgeInputingException


```
package com.vti.entity.accountQues11;

import com.vti.backend.InvalidAgeInputingException;
import com.vti.ultis.ScannerUltis;

public class AccountQues11 {
	public static int COUNT = 0;
	private int id;
	private String email;
	private String userName;
	private int age;

	@Override
	public String toString() {
		return "AccountQues11 [id=" + id + ", email=" + email + ",
userName=" + userName + ", age=" + age + "]";
	}

	public AccountQues11() throws InvalidAgeInputingException {
		super();
		System.out.println("Mời bạn nhập thông tin account cần tạo mới.
");
		this.id = COUNT++;
		System.out.println("Email: ");
		this.email = ScannerUltis.inputString();
		System.out.println("UserName: ");
		this.userName = ScannerUltis.inputString();
		this.age = inputAccountAge();
	}

	private int inputAccountAge() throws InvalidAgeInputingException {
		System.out.println("Age: ");
		while (true) {
			int ageInput = ScannerUltis.inputInt();
			if (ageInput <= 0) {
				throw new InvalidAgeInputingException("The age must
be greater than 0, please input again.");
			} else {
				if (ageInput < 18) {
					System.err.println("Your age must be greater
than 18, input again: ");
				} else {
					return ageInput;
				}
			}
```

```
		}

	}

}
```

➔Tạo main để demo

```
package com.vti.backend;

import com.vti.entity.accountQues11.AccountQues11;

public class Exercise2_Ques11_12 {
	public void question11_12() throws InvalidAgeInputingException {
		AccountQues11 accQues11 = new AccountQues11();
		System.out.println(accQues11.toString());

	}
}
```

Tạo main:

```
package com.vti.frontend;

import com.vti.backend.Exercise2_Ques11_12;
import com.vti.backend.InvalidAgeInputingException;

public class Demo_Exercise2_Ques11_12 {

	public static void main(String[] args) throws InvalidAgeInputingException
{
		Exercise2_Ques11_12 ex2_11_12 = new Exercise2_Ques11_12();
		ex2_11_12.question11_12();
	}

}
```