

Basic Programming Practicum

Jobsheet 5 Experiment



Name

Dicha Zelianivan Arkana

NIM

2241720002

Class

1i

Department

Information Technology

Study Program

D4 Informatics Engineering

1 Laboratory

1.1 Experiment 1

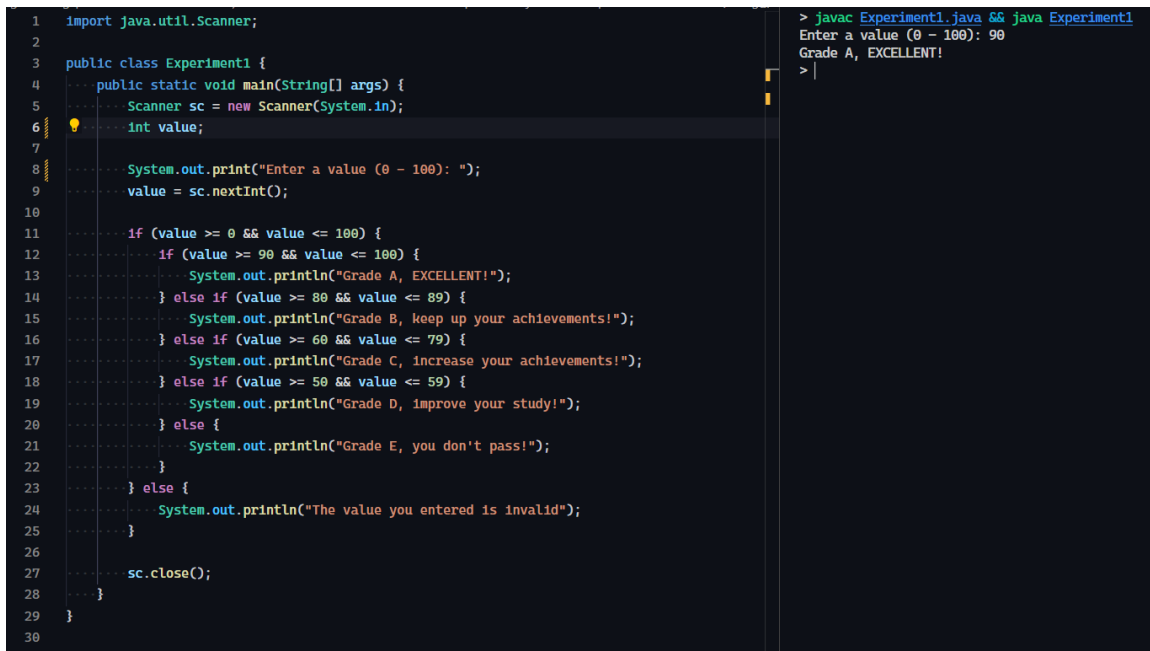
1. Open a text editor. Create a new file, name it **Nested1.java**
2. Write the basic structure of the Java programming language which contains the **main()** function
3. Add the **Scanner** library.
4. Make a **Scanner** declaration with the name **sc**
5. Create an **int** variable with the name **value**
6. Write down the syntax for entering the value from keyboard

```
System.out.print("Enter a value (0 - 100): ");
value = sc.nextInt();
```

7. Create a nested selection structure. The first check is used to ensure that the value entered is in the range 0 - 100. If the value is in the range 0 - 100, then a student graduation status will be checked, i.e. if the value is between 90 - 100 then the value is A, if the value is between 80 - 89 then the value is B, if the value is between 60 - 79 then the value is C, if the value is between 50 - 59 then the value is D, and if the value is between 0 - 49 then the value is E. Whereas if the value is outside the range 0 - 100 , then displayed information stating that the value entered is invalid.

```
if (value >= 0 && value <= 100) {
    if (value >= 90 && value <= 100) {
        System.out.println("Grade A, EXCELLENT!");
    } else if (value >= 80 && value <= 89) {
        System.out.println("Grade B, keep up your achievements!");
    } else if (value >= 60 && value <= 79) {
        System.out.println("Grade C, increase your achievements!");
    } else if (value >= 50 && value <= 59) {
        System.out.println("Grade D, improve your study!");
    } else {
        System.out.println("Grade E, you don't pass!");
    }
} else {
    System.out.println("The value you entered is invalid");
}
```

8. Compile and run the program. Observe the results!



```
1 import java.util.Scanner;
2
3 public class Experiment1 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         int value;
7
8         System.out.print("Enter a value (0 - 100): ");
9         value = sc.nextInt();
10
11         if (value >= 0 && value <= 100) {
12             if (value >= 90 && value <= 100) {
13                 System.out.println("Grade A, EXCELLENT!");
14             } else if (value >= 80 && value <= 89) {
15                 System.out.println("Grade B, keep up your achievements!");
16             } else if (value >= 60 && value <= 79) {
17                 System.out.println("Grade C, increase your achievements!");
18             } else if (value >= 50 && value <= 59) {
19                 System.out.println("Grade D, improve your study!");
20             } else {
21                 System.out.println("Grade E, you don't pass!");
22             }
23         } else {
24             System.out.println("The value you entered is invalid");
25         }
26
27         sc.close();
28     }
29 }
30
```

```
> javac Experiment1.java && java Experiment1
Enter a value (0 - 100): 90
Grade A, EXCELLENT!
> |
```

Figure 1: Experiment 1 Code and Output

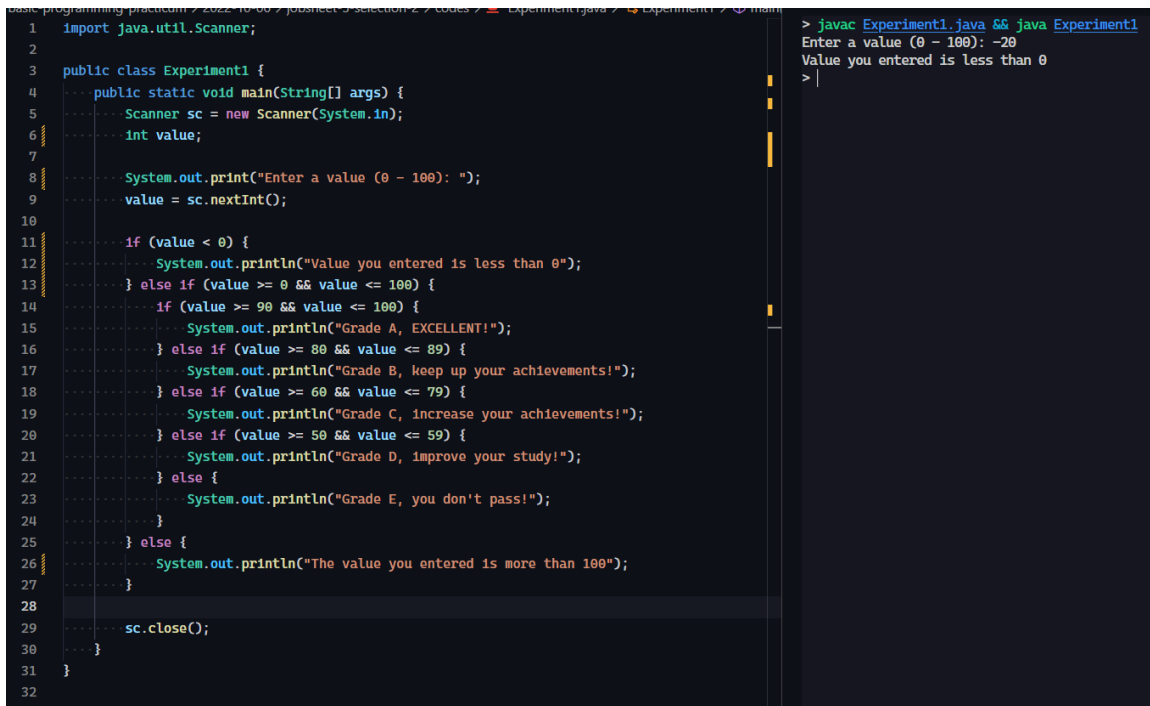
Questions

1. Describe the following syntax functions!

```
if (value >= 0 && value <= 100)
```

This syntax means that we're checking if `value` is greater than equal to 0 and it's less than equal to 100.

2. Modify the program code in Experiment 1 so that if the entered value is less than 0 the output "Value you entered is less than 0" and if the entered value is more than 100 the output will display "The value you entered is more than 100"!

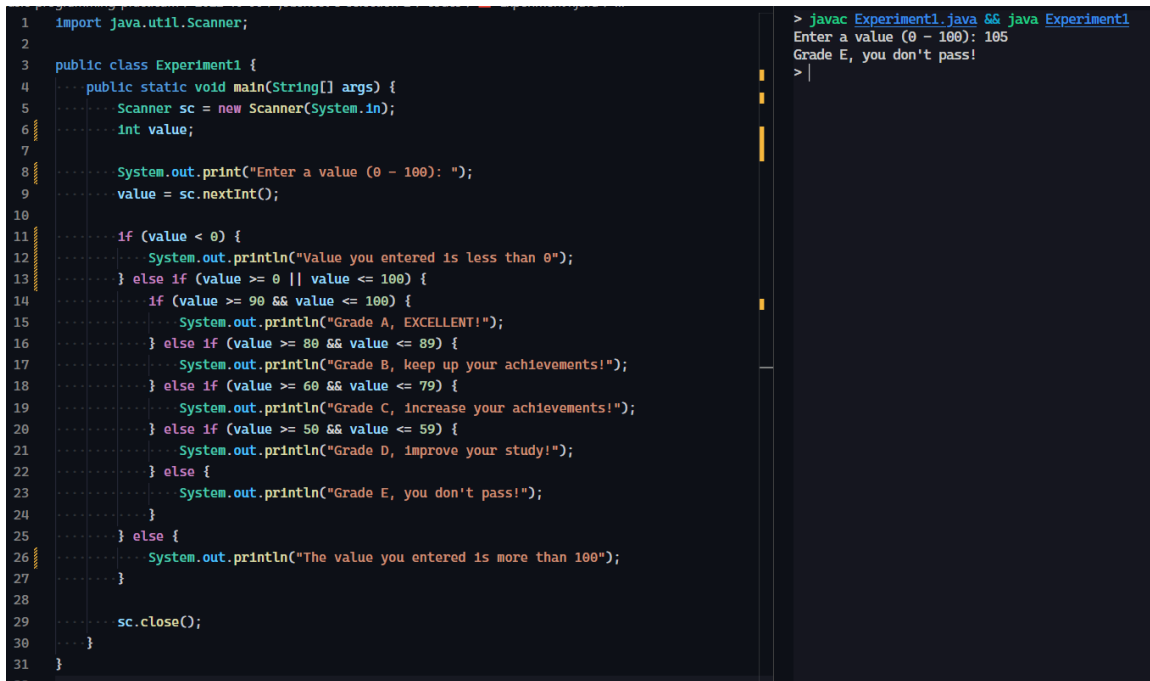


```
1 import java.util.Scanner;
2
3 public class Experiment1 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         int value;
7
8         System.out.print("Enter a value (0 - 100): ");
9         value = sc.nextInt();
10
11         if (value < 0) {
12             System.out.println("Value you entered is less than 0");
13         } else if (value >= 0 && value <= 100) {
14             if (value >= 90 && value <= 100) {
15                 System.out.println("Grade A, EXCELLENT!");
16             } else if (value >= 80 && value <= 89) {
17                 System.out.println("Grade B, keep up your achievements!");
18             } else if (value >= 60 && value <= 79) {
19                 System.out.println("Grade C, increase your achievements!");
20             } else if (value >= 50 && value <= 59) {
21                 System.out.println("Grade D, improve your study!");
22             } else {
23                 System.out.println("Grade E, you don't pass!");
24             }
25         } else {
26             System.out.println("The value you entered is more than 100");
27         }
28
29         sc.close();
30     }
31 }
32
```

```
> javac Experiment1.java && java Experiment1
Enter a value (0 - 100): -20
Value you entered is less than 0
> |
```

Figure 2: Experiment 1 Modified Code and Output

3. Change the `&&` operator to `||` on `if (value >= 0 && value <= 100)`. Compile and run the program by entering the value = 105 using keyboard. Watch what happened! Why is the result like that?



```
1 import java.util.Scanner;
2
3 public class Experiment1 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         int value;
7
8         System.out.print("Enter a value (0 - 100): ");
9         value = sc.nextInt();
10
11         if (value < 0) {
12             System.out.println("Value you entered is less than 0");
13         } else if (value >= 0 || value <= 100) {
14             if (value >= 90 && value <= 100) {
15                 System.out.println("Grade A, EXCELLENT!");
16             } else if (value >= 80 && value <= 89) {
17                 System.out.println("Grade B, keep up your achievements!");
18             } else if (value >= 60 && value <= 79) {
19                 System.out.println("Grade C, increase your achievements!");
20             } else if (value >= 50 && value <= 59) {
21                 System.out.println("Grade D, improve your study!");
22             } else {
23                 System.out.println("Grade E, you don't pass!");
24             }
25         } else {
26             System.out.println("The value you entered is more than 100");
27         }
28
29         sc.close();
30     }
31 }
```

```
> javac Experiment1.java && java Experiment1
Enter a value (0 - 100): 105
Grade E, you don't pass!
> |
```

Figure 3: Experiment 1 Modified Code and Output

Because the `||` condition checks if one of the expression is true. Since we set the value to be 105, the `value >= 0 || value <= 100` will be true. Because value is greater than 0. Even though it's greater than 100, one of the condition is already true. The final result is "Grade E, you don't pass!" because there is no branch that handles a value greater than 100 so it falls back to the default condition, which is inside the `else` branch.

1.2 Experiment 2

1. Observe the following flowchart!

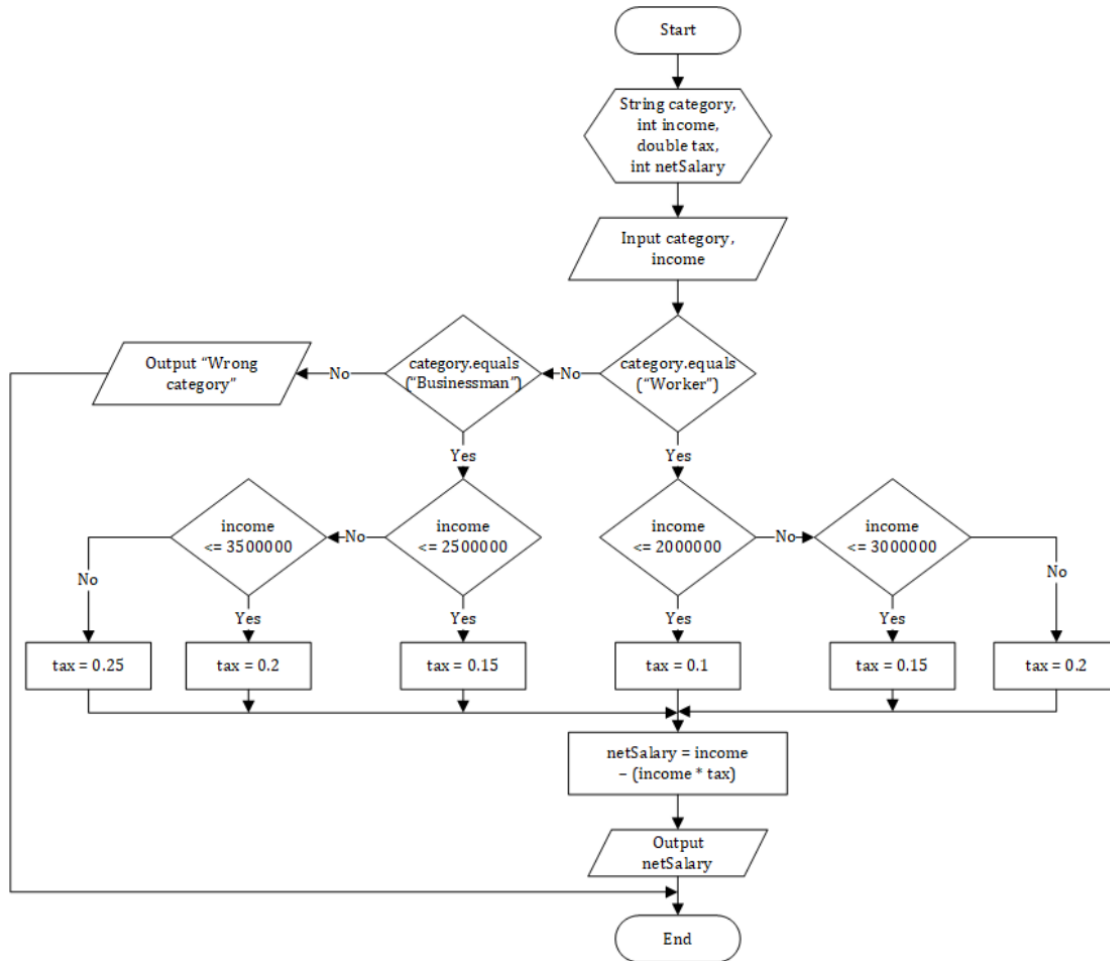


Figure 4: Experiment 2 Flowchart

The flowchart is used to calculate a person's net salary after taxes according to their category (worker and businessman) and the amount of income.

2. Open a text editor. Create a new file, name it **Nested2.java**
3. Write the basic structure of the Java programming language which contains the **main()** function
4. Add the **Scanner** library.
5. Make a **Scanner** declaration with the name **sc**

-
6. Declare `category`, `income`, `netSalary`, and `tax` variables

```
String category;  
int income, netSalary;  
double tax = 0;
```

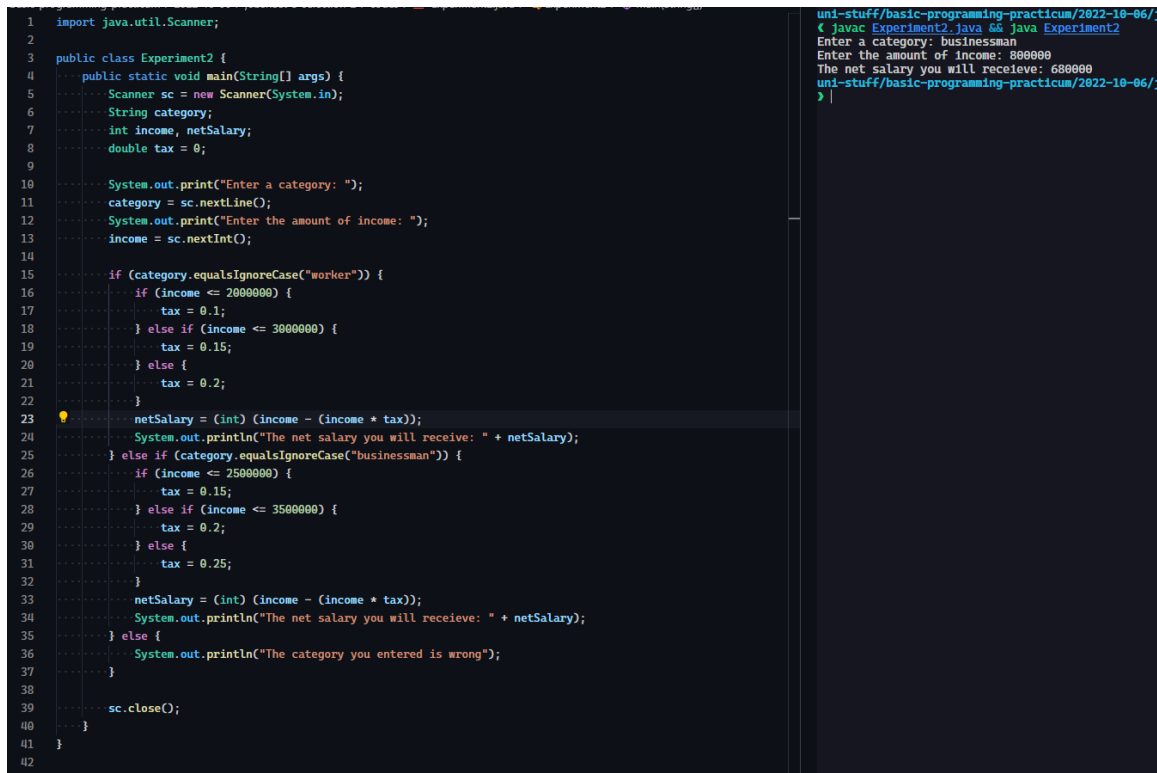
7. Write down the syntax for entering the value from keyboard

```
System.out.print("Enter a category: ");  
category = sc.nextLine();  
System.out.print("Enter the amount of income: ");  
income = sc.nextInt();
```

8. Create a nested selection structure. The first check is used to check the category (worker or businessman). Then a second check is carried out to determine the amount of tax based on the income that has been entered. Then add the program code to calculate the net salary received after taxes

```
if (category.equalsIgnoreCase("worker")) {  
    if (income <= 2000000) {  
        tax = 0.1;  
    } else if (income <= 3000000) {  
        tax = 0.15;  
    } else {  
        tax = 0.2;  
    }  
    netSalary = (int) (income - (income * tax));  
    System.out.println("The net salary you will receive: " + netSalary);  
} else if (category.equalsIgnoreCase("businessman")) {  
    if (income <= 2500000) {  
        tax = 0.15;  
    } else if (income <= 3500000) {  
        tax = 0.2;  
    } else {  
        tax = 0.25;  
    }  
    netSalary = (int) (income - (income * tax));  
    System.out.println("The net salary you will receive: " + netSalary);  
} else {  
    System.out.println("The category you entered is wrong");  
}
```

9. Compile and run the program. Observe the results!

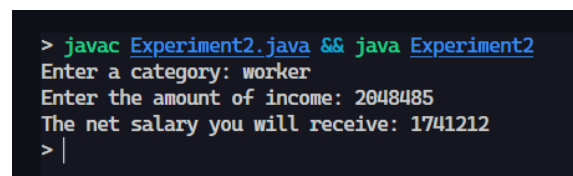


```
1 import java.util.Scanner;
2
3 public class Experiment2 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         String category;
7         int income, netSalary;
8         double tax = 0;
9
10        System.out.print("Enter a category: ");
11        category = sc.nextLine();
12        System.out.print("Enter the amount of income: ");
13        income = sc.nextInt();
14
15        if (category.equalsIgnoreCase("worker")) {
16            if (income <= 2000000) {
17                tax = 0.1;
18            } else if (income <= 3000000) {
19                tax = 0.15;
20            } else {
21                tax = 0.2;
22            }
23            netSalary = (int) (income - (income * tax));
24            System.out.println("The net salary you will receive: " + netSalary);
25        } else if (category.equalsIgnoreCase("businessman")) {
26            if (income <= 2500000) {
27                tax = 0.15;
28            } else if (income <= 3500000) {
29                tax = 0.2;
30            } else {
31                tax = 0.25;
32            }
33            netSalary = (int) (income - (income * tax));
34            System.out.println("The net salary you will receive: " + netSalary);
35        } else {
36            System.out.println("The category you entered is wrong");
37        }
38        sc.close();
39    }
40 }
41
42 uni-stuff/basic-programming-practicum/2022-10-06/
< javac Experiment2.java && java Experiment2
Enter a category: businessman
Enter the amount of income: 800000
The net salary you will receive: 680000
uni-stuff/basic-programming-practicum/2022-10-06/
> |
```

Figure 5: Experiment 2 Code and Output

Questions

1. Run the program by entering category = worker and income = 2048485 using keyboard. Watch what happened! Why is the decimal number not displayed?



```
> javac Experiment2.java && java Experiment2
Enter a category: worker
Enter the amount of income: 2048485
The net salary you will receive: 1741212
> |
```

Figure 6: Experiment 2 Output

The decimal number doesn't get displayed because we cast the final result back to integer. Specifically this part:

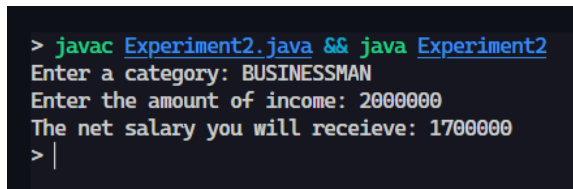
```
netSalary = (int) (income - (income * tax));
```


-
2. Describe the function of `(int)` in the following syntax!

```
netSalary = (int) (income - (income * tax));
```

It is used to calculate the net salary by subtracting the income with the tax and then casting the result to an integer using `(int)`.

3. Run the program by entering category = BUSINESSMAN and income = 2000000. Watch what happens! What are the uses of `equalsIgnoreCase`?



```
> javac Experiment2.java && java Experiment2
Enter a category: BUSINESSMAN
Enter the amount of income: 2000000
The net salary you will receive: 1700000
> |
```

Figure 7: Experiment 2 Output

The `equalsIgnoreCase` is used to compare two string ignoring the case sensitivity. Which explains why BUSINESSMAN is still equals to businessman

4. Change `equalsIgnoreCase` to `equals`, then run the program by entering category = BUSINESSMAN and income = 2000000. Watch what happens! Why is the result like that? What are the uses of `equals`?

The method `equals` compare two strings but it is case sensitive, while `equalsIgnoreCase` will ignore the case sensitivity. If we use `equals` to compare "businessman" with "BUSINESSMAN", it will be false but it will be true if we use `equalsIgnoreCase`.