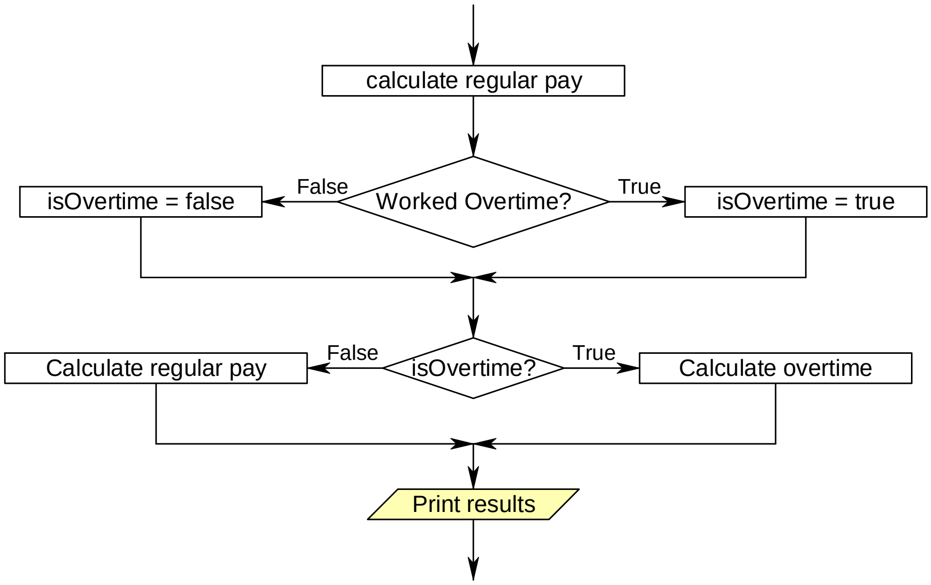
Virtual Lecture Notes (Part 2)

The Salary_v2 class is a minor variation of the Salary_v1 class. Both programs are designed to calculate an employee's salary based on the hourly pay rate and the number of hours worked. If the employee works more that 40 hours, time-and-a-half is paid for the overtime hours.

- Carefully study the source code for the Salary_v2 class you previously downloaded.
- Run the program and observe its performance and output.

Once again, the overall structure of the Salary_v2 class should seem very familiar to you after studying the AdmissionFeeV2 class. Can you spot the double branching conditional statements in the flowchart?



If you compare this to the flowchart for Salary_v1, you will notice some decision points where the flow of control branches based on evaluation of a boolean condition. Compare the flowchart with the corresponding code segment below.

```
…
<22>      boolean isOvertime = totalHours > 40;
<23>
<24>      if(isOvertime)
<25>         totalSalary=40*payRate+(totalHours-40)*payRate*1.5;
<26>      else
<27>         totalSalary=totalHours*payRate;
```

Line <22> declares `isOvertime` to be a `boolean` primitive data type and assigns `true` or `false` to the variable based on whether the total hours worked is greater than 40.

Line <24> evaluates the value of `isOvertime` (i.e., true or false). If the value is `true`, Line <25> is executed, otherwise the flow of control skips to Line <26>.

Line <25> calculates the total salary including both overtime and regular hours. Afterwards, the flow of control skips to Line <29>, the first print statement.

Line <26> is executed if `isOvertime` is `false` in Line <24>, then Line <25> is executed.

Line <27> calculates the total salary based only on regular hours worked.

Continue to analyze the program line-by-line and make sure you understand the syntax and purpose of each statement.

Although both versions of this program work equally well, saving two lines of code with the simple **`if`** statement does not compensate for its lack of clarity. Until you attain expert status as a programmer, it is better to write code that can be clearly understood, even if it means typing a few extra lines. Consequently, **`if-else`** is a better choice in this situation.