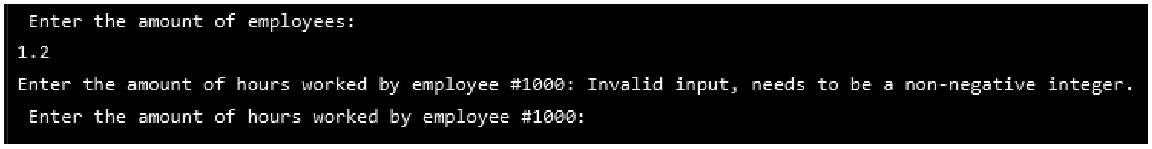
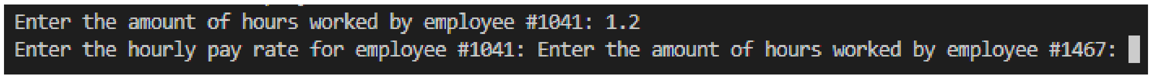
Group Report: Module 3, Payroll Calculator
Group C: Toby Hansen, Dowon Yang, Rakin Bhuiyan
Group Leader and Reporter: Toby Hansen
February 24, 2025

Unfortunately, out of my two group mates, Rakin Bhuiyan hasn't gotten in contact with us at all, and Dowon Yang responded to my forum post four hours before the deadline. Rakin also did not post an introduction at the beginning of the semester, so I'm not entirely sure whether or not he's participating in the class. Dowon suggested that we discuss our solutions today and submit the refined version tonight (day after the deadline), but I didn't want to submit it any later than absolutely necessary, so I submitted my personal solution last before refining it with our group. Given that I still have yet to receive feedback from Dowon in the forum, and Rakin still hasn't appeared, I am taking it upon myself to write the group report and submit it before the end of the night. My improvisation will include a focus on issues I discovered with my own program.

The first issue I discovered while QA testing was that my edge case for detecting invalid input when it came to integers did not work when a decimal was entered instead.

1. 

```
 Enter the amount of employees:
1.2
Enter the amount of hours worked by employee #1000: Invalid input, needs to be a non-negative integer.
 Enter the amount of hours worked by employee #1000:
```

2. 

```
Enter the amount of hours worked by employee #1041: 1.2
Enter the hourly pay rate for employee #1041: Enter the amount of hours worked by employee #1467:
```

In both of these cases, when I entered 1.2 (a number containing a decimal), the program accepted the input and assigned 1 to the initial integer, then attempted to assign .2 to the next variable. In the first case, 'amount of employees' was assigned 1, and then .2 was attempted to be assigned to 'amount of hours' but because there was a decimal before the two it was registered as invalid and triggered the invalid sequence. In the second case, 'amount of hours' was assigned 1, and then .2 was assigned to 'hourly pay rate'. For some reason, the program requests 'amount of hours' for a second time after the 'hourly pay rate' is requested, but either way this bug needed to be fixed by including extra edge case protection. Originally, I only had a while loop checking if the cin was able to properly be assigned to the intended variable, and if it wasn't, it would tell the user that the input was invalid ask them to input it again. To be as safe as possible, it would be best to assign the cin to a temporary variable and check against it. A simple if statement can deduce whether the input is a decimal, and if not it goes through the original while loop check.

The second issue I discovered was just now as I am writing this report. I am now realizing that the instructions specifically mention "No use of conditional statements to handle edge cases." This throws my entire edge case handling system out of the window. In order to fix this issue, a simple try-catch statement can be used each time data is requested from the user. The program 'tries' to assign the input to a variable, and if it doesn't work we 'catch' it and ask them to re-enter the input.

This concludes the report of issues in my program that I discovered throughout testing. As the group leader I apologize for our disorganization, and I hope that going forward we are able to talk and work together.

Thank you for reading,
Toby Hansen