# Payroll Pseudocode Final Project

Course: SDEV 120

Professor: Prof. Mike Taylor

Group Members: Brett Day, Adrian Gallegos, Danielle Main, Alexander Perez, Eden Marshall, Alexander Tellef, Garrett Williams, Nickolas Lehman

Date: August 2nd, 2025

# Introduction

This group project allows us to create a comprehensive payroll system using pseudocode to track and validate employee working hours. This program demonstrates time entry capabilities, validation of regular and overtime hours, automatic overtime detection, and it also generates detailed time tracking reports. This program ensures accurate and fair compensation for employees within an organization, while maintaining compliance with regulatory requirements and labor regulations.

The entire body of the pseudocode is structured in an easy-to-follow way below, and it is filed into multiple different logical sections, each one handling a specific part of the payroll process. This program is also designed to be modular, easy to maintain, and easily expandable for future enhancements.

## Time Entry Functions

This block of logic serves to prompt the user for data input, which in this case is hours worked by each employee for each day of the week. This logic also serves to store the data for later validation.

1time entry function

TIME ENTRY FUNCTIONS

Begin Program

Function GetHoursWorked (employee)

INIT hours\_list = empty linst

For each day in week Do

Prompt “ Enter hours worked by [employee] on [day]:

Input daily\_hours

Add daily hours To hours\_list

End FOR

Return hours\_list

End function

## Hours Validation Logic

This function calculates the total hours worked and separates them into regular and overtime hours if the total exceeds 40 hours.

2 hours validation logic

HOURS VALIDATION LOGIC

Function validateHours(hours\_list)

total\_hours = sum(hours\_list)

if total\_hours > 40 then

regular\_hours = 40

## Overtime Detection Algorithm

This logic identifies whether an employee has worked overtime and determines how many overtime hours need to be compensated for.

3overtime detection algorithm

4 timetracking reports

5 (4) test cases

overtime\_hours = total\_hours - 40

end If

else

regular\_hours = total\_hours

overtime\_hours = 0

end else

return

“ Total\_hours :” total hours, regular\_hours;” regular\_hours,

“Overtime\_hours:” overtime\_hours

End function

Print “Overtime Hours: “, result.overtime\_hours

If result.overtime\_hours > 0

Print “Overtime Status : YES”

Else

Print “Overtime Status : NO”

End if

End function

## Generate Report

This function produces a weekly time report summarizing total hours, regular hours, overtime hours, and whether the employee qualifies for overtime pay.

GENERATE REPORT:

Function GenerateReport(employee, result)

Print “Wekly Time Report for: “, result.total\_hours

Print “Total Hours Worked: “, result.total\_hours

Print “Regular Hours: “, result.regular\_hours

GenerateReport(result) // generate report

End FOR

End program

## Main Interface

This is the main execution logic that ties together the time entry, validation, and reporting functions for each employee.

MAIN INTERFACE:

Begin program

For each employee IN employee\_list DO

Hours\_list = GetHoursWorked(employee) // link first function and get time

Result = validateHours(hours\_list) // validate time

# Test Cases

The following test cases were used to verify the correctness of the payroll pseudocode.

|  |  |  |  |
| --- | --- | --- | --- |
| Employee | Input Hours | Expected Regular Hours | Expected Overtime Hours |
| Alice | 8,8,8,8,8 | 40 | 0 |
| Bob | 10,10,10,5,5 | 40 | 0 |
| Charlie | 9,9,9,9,9 | 40 | 5 |
| Dana | 12,12,12,12,12 | 40 | 20 |

# Conclusion

This logic demonstrates that we created a program that is capable of collecting working hours, validating those hours, and compiling that data into reports. Since we broke down the project into logical blocks of pseudocode, we were able to create a payroll system that is accurate and efficient. Future iterations of this program could include live database integration, GUI interfaces, and better reporting capabilities.