# Lab 2 – Computing Taxes and Pay

## Part I NetPay

Create a class called **NetPay.java** to compute a person's gross and net pay based on their hourly wage, hours worked, and several withholdings. All statements should be defined in the **main** method of the class (except for declarations of constants).

The output of the program should look similar to this:

```
Hours per Week:         40
Gross Pay:              290.0
Net Pay:                225.765

Deductions
Federal:                29.0
State:                  13.05
Social Security:        17.98
Medicare:               4.205
```

Implementation Details:

- To make the outputs line up on the right-hand side, you will need to use the tab character '\t' in your print statements. You may need one or more '\t' characters in each line. Play around with it until the output lines up.
- You will use the number of hours per week to compute the gross pay, net pay, and the deductions. You will need to create a variable for each value above. Brainstorm the proper data types and include that in your writeup.
- Hours per week is your input to the program. For Part I, you will type it into the program. For Part II (below), it will come from the keyboard.
- You will also need to declare named constants and initialize them to the values shown.
  - **FEDERAL_TAX_PERCENT**:   10.00.
  - **STATE_TAX_PERCENT**:    4.5.
  - **SS_PERCENT**:    6.2.
  - **MEDICARE_PERCENT**    1.45.
  - **PAY_PER_HOUR**    7.25.

Questions to consider for the writeup:
- How do we compute gross pay? Store the result in the variable created for gross pay.
- How do we compute federal tax? Think about using gross pay and federal tax percent.
- Compute state tax, social security contribution, and medicare using similar methods as federal tax.
- How do we compute net pay?
- Display the values on the screen in the order listed above.

Once your program is implemented, compile and run your program. Afterwards, edit your program by changing the value assigned to the variable for hours per week. Additional sample executions are shown below.

```
Hours per Week:      30
Gross Pay:           217.5
Net Pay:             169.32375

Deductions
Federal:             21.75
State:               9.7875
Social Security:     13.485
Medicare:            3.15375
```
```
Hours per Week:      35
Gross Pay:           253.75
Net Pay:             197.544375

Deductions
Federal:             25.375
State:               11.41875
Social Security:     15.7325
Medicare:            3.679375
```
```
Hours per Week:      40
Gross Pay:           290.0
Net Pay:             225.765

Deductions
Federal:             29.0
State:               13.05
Social Security:     17.98
Medicare:            4.205
```
```
Hours per Week:      45
Gross Pay:           326.25
Net Pay:             253.985625

Deductions
Federal:             32.625
State:               14.68125
Social Security:     20.2275
Medicare:            4.730625
```

## Part II Modification of NetPay

Modify **NetPay.java** to **ask the user for the number of hours they have worked for** (it is no longer typed into the program). *Note: you do not need to submit the code for Part I – only submit a single java file after completing both parts.*

- Which preexisting Java class did we use to read in keyboard input? Reference the textbook or class slides and borrow the necessary code from there (you don't have to memorize this code).
- Don't forget to prompt the user to enter the number of hours per week.
- What should you do to validate users' inputs? What if a user inputs a negative value?

Compile and run your program. A run of your program should look like this (**the blue (bold) is the user's input – it is not being output by the program**):

```
Hours per Week:        100
Gross Pay:             725.0
Net Pay:               564.4125

Deductions
Federal:               72.5
State:                 32.625
Social Security:       44.95
Medicare:              10.5125
```

You should run your program with various inputs, and test that your program is producing correct outputs. For every programming lab and project this semester, you are responsible for ensuring that your program doesn't contain any syntax, runtime, or logical errors. Always test, test, and retest the programs you create.

## Submission and Grading

It is required that you organize this lab in a single subfolder named '*lab2*' under the cloned GitLab project folder, and synchronize everything to the GitLab project you created previously by periodical committing/pushing your local changes to GitLab. **30 points would be deducted if there are no commit/push in your GitLab repository before 11:59 pm, Oct 12th.**

Please use comments to explain your program or any sections that are possibly difficult to understand. **10 points would be deducted if few or no comments can be found in your java files**.

The lab will be graded according to the following guidelines.

- A score between 0 and 100 will be assigned.
- If the source file(s) are not submitted by the deadline, then a grade of 0 will be assigned.
- 10 points will be deducted for late labs per day.
    - o 0-24 hours late -10
    - o 24-48 hours late -20
    - o >48 hours late = 0

After you have completed and thoroughly tested your program, upload and submit **NetPay.java** (**not** the .class file – that's the byte-code) to *Canvas*. Always double check that your submission was successful on *Canvas*! *Note: you do not need to submit the code for Part I – only submit a single java file after completing both parts.*