**PFC - Assignment – part 1**

**Income tax calulator   
with Functions and Robus validation**

As your first assignment this semester, you are to write a C program that calculates the income tax .

**LEARNING OUTCOME**

Upon successful completion of this assignment, you will have demonstrated the ability to use sequence, selection, iteration constructs as well as functions.

**INTRODUCTION**

From 2009, people must pay income tax depended on the total amount he/she earns. Income tax is the kind of progress tax, which applies higher tax rates to individuals posting higher earnings.

Firstly, taxable income is identified. Taxable income is calculated by deducting total deduction (personal allowance and any deductible reliefs you are entitled to) from your total earnings.

Here, five kinds of deductions:

1. 4.000.000 VND per month for yourself
2. 1% of total amount for health insurance
3. 5% of total amount for pension contribution
4. 1.600.000 VND for each dependant under 18 age
5. any gift of charity

Then, taxable income is multiplied by the tax rate of according brackets you belong in.

Here, seven tax brackets:

1. taxable income of **5 millions VND or less** attracts tax at

**5%**

1. taxable income of **more than 5 millions and not more than 10 millions** attracts tax at **10%**.
2. taxable income of **more than 10 millions and not more than 20 millions** attracts tax at **15%**.
3. taxable income of **more than 20 millions and not more than 30 millions** attracts tax at **20%**.
4. taxable income of **more than 30 millions and not more than 50 millions** attracts tax at **25%**.
5. taxable income of **more than 50 millions and not more than 80 millions** attracts tax at **30%**.
6. taxable income of **more than 80** attracts tax at

**35%**.

**SPECIFICATIONS**

Your program prompts for and accepts the following information:

* The income earned by the employee (including pension contribution and health insurance or not)
* Number of dependants under the age of 18
* Authourized deductions (any gift of charity…)

**TECHNICAL NOTES**

**Conversion Specifiers**

By default, the **%lf** printf conversion specification displays six (6) decimal places without leading spaces.  To limit the number of decimal places to two (2), use **%.2lf** instead of **%lf**. To display the number right-justified in a field of 10 columns, use **%10.2lf**.  This will align the decimal points on your output.

**Program Constants**

Design your program so that all of the constants for 2009 can be changed for subsequent years with minimal effort and a single recompilation. E.g., you can define 'constants' that may change from year to year, use the preprocessor directive:

**#define SYMBOLIC\_NAME VALUE**

Place this directive before your main program and after your header comments.  Use the SYMBOLIC\_NAME within your program code and set the VALUE in the preprocessor statement.  For example, for 2009

**#define RATE\_PENSION 0.05**

**Then statements for calculating the deduction for pension would be as below:**

**double income, p;**

**printf("Enter income : ");**

**scanf("%lf", &income);**

**p = income \* RATE\_PENSION;**

**printf("Deduction for pension is %10.2lf\n", p);**

**FUNCTION DESIGN**

Design your program so that another programmer can modify it with minimal intrusion.  Divide your code into functions to achieve the highest possible cohesion and the lowest possible coupling.  For guidelines on designing functions see the [Modularity](http://cs.senecac.on.ca/~btp100/pages/content/modul.html) reading.

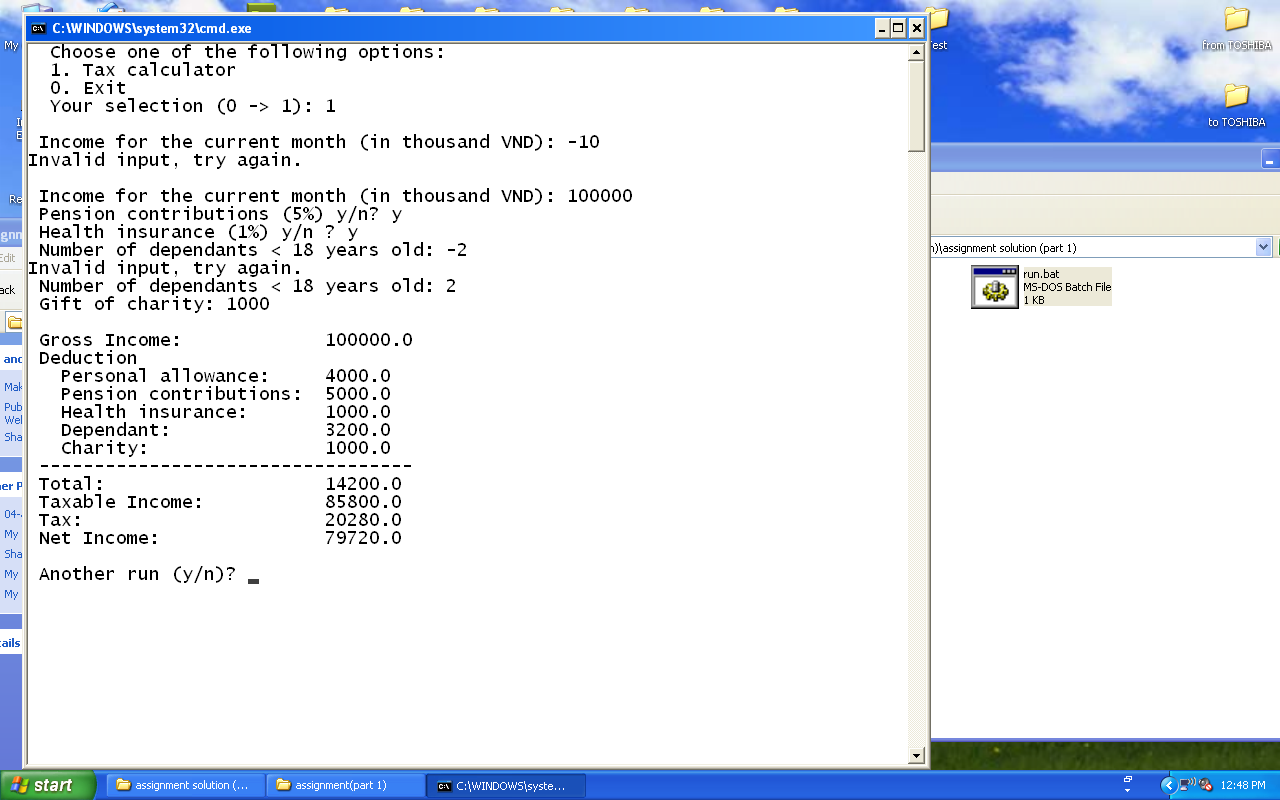
Your program asks the user whether or not to continue with another run, and if the user answers yes, your program continues without terminating execution.

**ROBUST VALIDATION**

Your program accepts and validates any input whatsoever from the user.  DO NOT assume that the user will only enter numbers, white space or possibly a decimal point, but no other characters.  Your program generates an appropriate response for any erroneous input.  Examples of unacceptable or unreasonable input include:

* input is negative in value
* more than one decimal point appears in floating-point input
* characters are embedded in numeric input
* character is different from ‘y’ and ‘n’
* gift of charity exceeds the income
* a decimal point appear in integer input

Your output program may look something like:



You can run the **tax1.exe** file to know how the program should work.

**SUBMISSION REQUIREMENTS**

At the beginning of your program, include comments reflecting accurate information for you in the format as below:

**/\***

**Assignment 1 - Income tax**

**Class ID : SE0412**

**Student ID : 00400**

**Student Name : Nguyễn Minh Đức**

**Due Date : 20 Decmber 2009**

**I declare that this assignment is my own work**

**in accordance with FPT Policy.**

**\*/**