Same submission requirement as before. Your program must contain proper documentation. Every function must have function documentation header (see example from printMeFirst() function.

You must separate .h file from .cpp file. You must use Makefile. You .h file must use safeguard (use #ifndef ….. #endif)

Below is details of this lab:

Bank Accounts

Implement a class Account. The class object account has:

- a data member balance (double balance)
- class member functions to
  - void deposit (double amount);
  - void withdraw (double amount);
  - double getBalance() const;

Implement a class Bank. This bank has two class data member, Account checking and Account savings, of the type Account. Implement member functions:

```
 class Bank
{
public:
   Bank();
   Bank(double checkingAmount, double savingsAmount);
  // deposit member function will call Account member function deposit
   void deposit(double amount, string account
   void withdraw(double amount, string account);
   void transfer(double amount, string account);
   void printBalances() const;

private:
   ????? checking;   // change ????? with correct data type for variable checking
   ??????? savings;  // use correct data type for variable savings
};
```

Charge a $5 penalty if an attempt is made to withdraw more money than what is available in the account. For example, if your checking account balance is 1000 and you want to withdraw 1500, then your program should not allow the withdraw, in addition, you need to change a over drawn penalty. So you balance would be $1000 - $5 = $995.

Here the account string is “S” or “C”. “S” is for savings account. “C” is for checking account. For the deposit or withdraw, it indicates which account is affected. For a transfer it indicates the account from which the money is taken;the money is automatically transferred to the other account.

Use global CONSTANT for penalty (PENALTY). Don’t hard code the number in your code.

You may need to add more member variables and functions to the classes thanthose listed above, and you need to implement these member functions.

You must store class Account and class Bank in a .h file. You must use safe guard (#ifndef) in the .h file. Implementation of the class member functions mustbe in a separate cpp file.

You need to use Makefile to compile the programs.

Use the c++ main program file below to run the program.

```
int main()
{

   printMeFirst("Ron Sha", "CS-116 - 2021 Spring");

   Bank my_bank;
   cout << "\nInital bank balances: \n";
   my_bank.print_balances();   /* set up empty accounts */

   cout << "\nAdding some money to accounts: \n";

   cout << "\nAdding $1000 to saving \n";
   cout << "Adding $2000 to checking \n";
   my_bank.deposit(1000, "S");  /* deposit $1000 to savings */
   my_bank.deposit(2000, "C");  /* deposit $2000 to checking */
   my_bank.print_balances();

   cout << "\nTaking out $1500 from checking,and moving $200 from";
   cout << " savings to checking.\n";

   my_bank.withdraw(1500, "C"); /* withdraw $1500 from checking */
   my_bank.transfer(200, "S");  /* transfer $200 from savings */
   my_bank.print_balances();

   cout << "\ntrying to withdraw $900 from Savings.\n";
   my_bank.withdraw(900,"S");
   my_bank.print_balances();

   cout << "\ntrying to withdraw $400 from Checking.\n";
   my_bank.withdraw(400,"C");
   my_bank.print_balances();

   return 0;
}
```

```
int main()
{

   printMeFirst("Ron Sha", "CS-116 - 2021 Spring");

   Bank my_bank;
   cout << "\nInital bank balances: \n";
   my_bank.print_balances();   /* set up empty accounts */

   cout << "\nAdding some money to accounts: \n";

   cout << "\nAdding $1000 to saving \n";
   cout << "Adding $2000 to checking \n";
   my_bank.deposit(1000, "S");  /* deposit $1000 to savings */
   my_bank.deposit(2000, "C");  /* deposit $2000 to checking */
   my_bank.print_balances();


   cout << "\nTaking out $1500 from checking,and moving $200 from";
   cout << " savings to checking.\n";

   my_bank.withdraw(1500, "C"); /* withdraw $1500 from checking */
   my_bank.transfer(200, "S");  /* transfer $200 from savings */
   my_bank.print_balances();


   cout << "\ntrying to withdraw $900 from Savings.\n";
   my_bank.withdraw(900,"S");
   my_bank.print_balances();

   cout << "\ntrying to withdraw $400 from Checking.\n";
   my_bank.withdraw(400,"C");
   my_bank.print_balances();


   return 0;
}
```

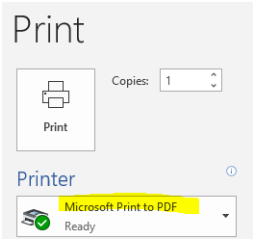
Your program output should be similar to the one below:

```
Program written by: Ron Sha
Course Info: CS-116 - 2021 Spring
Date: Mon Feb 22 10:35:03 2021


Inital bank balances:
Savings account balance:  $   -0.00
Checking account balance: $    0.00

Adding some money to accounts:

Adding $1000 to saving
Adding $2000 to checking
Savings account balance:  $ 1000.00
Checking account balance: $ 2000.00

Taking out $1500 from checking,and moving $200 from savings to checking.
Savings account balance:  $  800.00
Checking account balance: $  700.00

trying to withdraw $900 from Savings.
Only 800.00 is available.  But tring to withdrawn 900.00.  Deduct 5 from account
Savings account balance:  $  795.00
Checking account balance: $  700.00

trying to withdraw $400 from Checking.
Savings account balance:  $  795.00
Checking account balance: $  300.00
```

Lab Submission: See the lab submission requirements published in canvas.

To submit your assignment in canvas, you must submit TWO files (one pdf and one zip) as follows:

1. **Attach pdf file** which contains source codes you have written and program output screenshot so I can easily read in one file. You can use a word editor to place all the required programs and screenshots, and then use 'Print' to 'Microsoft Print to PDF' to save to a pdf file



The pdf file MUST have the following sections (1. Program Description, 2. Program Source Code and 3. Program Output).

1. Program Description

- brief *description* of the purpose of the *program and*
- an explanation of what your software does and what problem it solves

2. Program Source Code

- Include all the source codes (program files) you have written for this lab (screenshots are ok)
- Your program must have adequate documentation for your source codes:
  - Program description – see above on Program Description
  - **You must put the following function headers for each function (the function header MUST be placed just above the function declaration in your source code). For Functions, it must have:**
    - *Function name: name of this function*
    - *Function description: the purpose of this function and how to use it*
    - *@param param_name and what the parameter/argument is used for*

- *@return what is returned from the function*

- Separate all .h (definition files) with .cpp (implementation files).
- Make sure your screenshots are readable (not too small)

Below is an example of source code

File: print_me_first_main.cpp (list the program file individually)

```cpp
//print_me_first_main.cpp
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

/**
 * @Purpose - this function print out the person who wrote the program,
 * and date/time the program run.
 * @parm - name - the author of the program
 * @parm - courseInfo - the name of the course
 * @return - none
 * @author - Ron Sha
 **/

void PrintMeFirst(string name, string courseInfo)
{
    cout <<" Program written by: "<< name << endl; // put your name here
    cout <<" Course info: "<< courseInfo << endl;
    time_t now = time(0); // current date/time based on current system
    char* dt = ctime(&now); // convert now to string for
    cout << " Date: " << dt << endl;
}
```

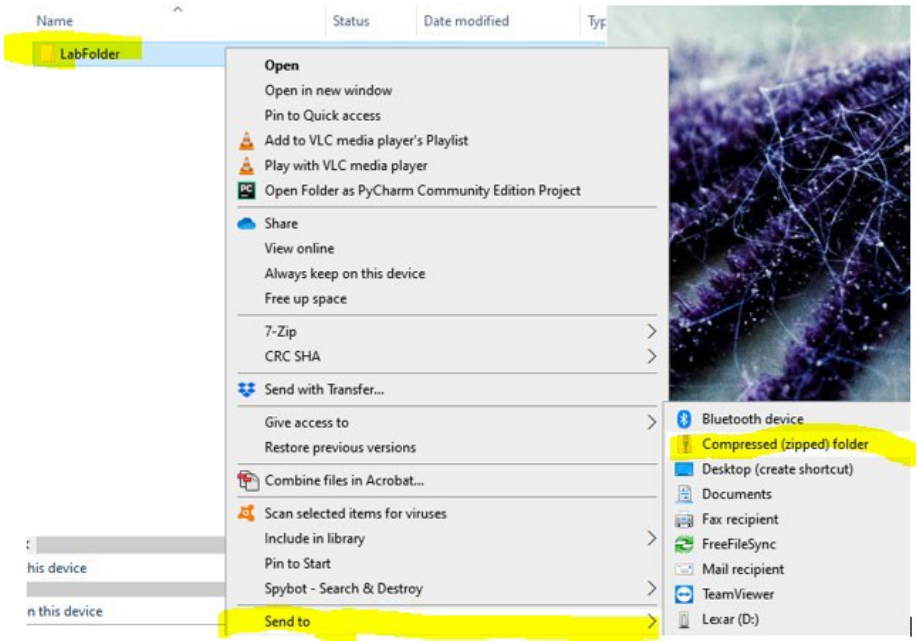
File: list other cpp files separately

3. Program Output

- Attach all the program outputs (screenshots)
- Don't place Source code and program output side by side as it is not readable in screenshot
- Make sure your screenshots are readable (not too small)
- Your main program outputs MUST include your name printout (use the print_me_first function/program).

2. **Attach zip file** which contains all your source code (you can zip the folder) and functions. Even if you only have one source file, you MUST still do a zip file of the folder. I must be able to compile and run your program from all the source code programs after I unzip your zip file.

   You should create a folder for each lab, and place all your programs, functions and all other files related to this lab in this lab folder. To submit the lab folder, you can use "Send to -> compress" in window file explorer to create a zip file of the folder.



3. **Now you can upload both the pdf file and zip file separately as your assignment submission in canvas.**