**C++ Part I (INFO1-CE9264) Fall 2014 – Homework 1**

Clement Chan

**Question 1:**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

#define METRICTONS 35273.92

int main(){

float wcerealounces;

float wcerealmt;

float noboxes;

cout << "Enter cereal weights in ounces: " << endl;

cin >> wcerealounces;

wcerealmt = wcerealounces/METRICTONS;

noboxes = ceil(wcerealmt);

cout << "The weight of cereal in ounces is: " << wcerealounces << endl;

cout << "The weight of cereal in Metric Tons is : " << wcerealmt << endl;

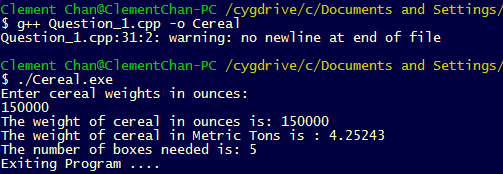
cout << "The number of boxes needed is: " << noboxes << endl;

cout << "Exiting Program .... " << endl;

return 0;

}

**Output:**



**Question 3:**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

//Define a class that includes all the calculations

class Wagecalc{

private:

float oldsalary;

float Rate;

public:

float newretro(float, float);

float newannual(float, float);

float newmonthly(float, float);

};

// Methods of the functions included in the class

float Wagecalc::newretro(float a, float b){

double newrwage;

oldsalary = a;

Rate = b;

newrwage = oldsalary\*(1+Rate/2)-oldsalary; //The wage increase only occurs for 6 months

return newrwage;

}

float Wagecalc::newannual(float a, float b){

float newawage;

oldsalary = a;

Rate = b;

newawage = oldsalary\*(1+Rate/2); //The new annual salary with the 6 months retro increase

return newawage;

}

float Wagecalc::newmonthly(float a, float b){

float newmwage;

oldsalary = a;

Rate = b;

newmwage = (oldsalary\*(1+Rate/2))/12; //The new monthly salary with the 6 months retro increase

return newmwage;

}

//Main class that deliver all the calculations

int main(){

Wagecalc WC;

const float payincrease = 0.076;

float salary\_input;

cout << "Please enter your salary last year: " << endl;

cin >> salary\_input;

cout << "Amount of retroactive pay for 6 months is: " << WC.newretro(salary\_input, payincrease) << endl;

cout << "New annual salary of employee is : " << WC.newannual(salary\_input, payincrease) << endl;

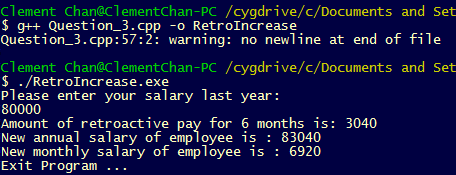
cout << "New monthly salary of employee is : " << WC.newmonthly(salary\_input, payincrease) << endl;

cout << "Exit Program ... " << endl;

return 0;

}

**Output:**



**Question 4:**

**Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Logic Understanding--

FV = Face Value

R = interest Rates

T = Time in years

CR = Consumer receivables

FV - (FV \* R \* T) = Consumer Received

FV \* (1- R\*T) = Consumer Received

FV = Consumer Received / (1 - R\*T);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <cmath>

using namespace std;

class FVRepayCalc{

private:

float R;

float T;

float CR;

public:

float FVcalc(float, float, float);

float Repayment(float, float);

};

float FVRepayCalc::FVcalc(float a, float b, float c)

{ R = a;

T = b;

CR = c;

float FV = CR/(1-(R/100\*T));

return FV;

}

float FVRepayCalc::Repayment(float a, float b)

{

float FV = a;

float T = b;

float Tmonth = T \* 12;

float RV = FV/Tmonth;

return RV;

}

int main(){

FVRepayCalc FRC;

float interest;

float time;

float ConsumerReceive;

float futurevalue;

cout << "Enter Interest in %: " << endl;

cin >> interest;

cout << "Enter Duration in Years: " << endl;

cin >> time;

cout << "Enter amount to receive: " << endl;

cin >> ConsumerReceive;

futurevalue = FRC.FVcalc(interest, time, ConsumerReceive);

cout << "The Future Value is: " << futurevalue << endl;

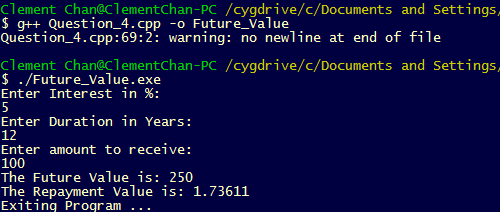
cout << "The Repayment Value is: " << FRC.Repayment(futurevalue, time) << endl;

cout << "Exiting Program ... " << endl;

return 0;

}

**Output:**



**Question 7:**

**Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[In] : Pounds, Minutes, Activities

[Out] : Total Calories Burnt for each Activity

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <cmath>

using namespace std;

inline float TotalCal(float weight, float time, int METS){

float TotalCalories;

TotalCalories = 0.0175 \* weight \* time \* METS;

return TotalCalories;

}

inline float WKilo(float weightpound){

float weightkil;

weightkil = weightpound/2.2;

return weightkil;

}

int main() {

float weightpound;

float weightkilo;

float minutes;

cout << "Enter your weights in pound: " << endl;

cin >> weightpound;

cout << "Enter exercise time in minutes: " << endl;

cin >> minutes;

weightkilo = WKilo(weightpound);

cout << "Total Calories Burned For Running 6 MPH is: " << TotalCal(weightkilo,minutes, 10) << endl;

cout << "Total Calories Burned For Basketball is: " << TotalCal(weightkilo,minutes, 8) << endl;

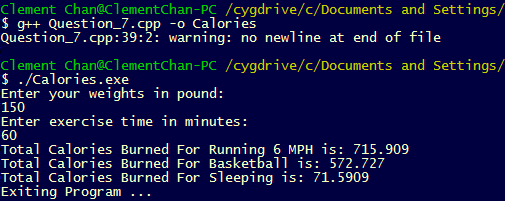
cout << "Total Calories Burned For Sleeping is: " << TotalCal(weightkilo, minutes, 1) << endl;

cout << "Exiting Program ... " << endl;

return 0;

}

**Output:**



**Question 8:**

**Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[in] : n, Guess\_Number

[out] : Two decimals double estimation

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <iomanip>

#include <stdio.h>

using namespace std;

inline double Babylonian(double n, double guess, int i){

if (i <= 5) {

double r = n/guess;

guess = (guess + r)/2;

return guess;

};

}

int main() {

double number;

double Guess\_Number;

cout << "Enter the positive number n: " << endl;

cin >> number;

if (number < 0){

printf ("n is negative. The Babylonian cannot be applied.\n");

}

else{

Guess\_Number = number/2; //Initial guess on the numbers

for (int i = 1; i <= 5; i++){

Guess\_Number = Babylonian(number, Guess\_Number, i);

};

cout << "The square root of n using Babylonian algorithm is: " << setprecision(3) << Guess\_Number << endl;

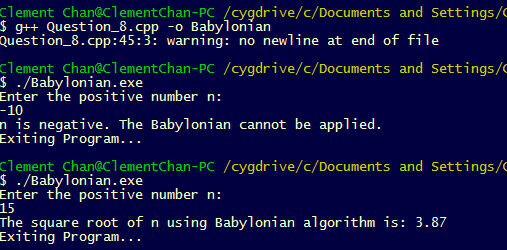
};

cout << "Exiting Program... " << endl;

return 0;

};

**Output:**



**Question 10:**

**Code:**

#include <iostream>

#include <math.h>

using namespace std;

int main(){

float acceleration;

float distance;

float time;

cout<<"Please enter the time of freefall in seconds: " << endl;

cin >> time;

acceleration = 32;

distance = 0.5 \* acceleration \* time \* time;

cout<<"The distance given the time you entered in second is: " << distance << endl;

cout<<"Exiting Program ... "<<endl;

return 0;

}

**Output:**

