**C++ Part I (INFO1-CE9264) Fall 2014 – Homework 7**

Clement Chan

**Question 1 – Month**

**Header file**

class Month{

private:

int month;

public:

Month(); //Default Constructors

Month(char\*); // Constructors

Month(int); // Constructors

char\* map\_char(int);

void map\_int(char\*);

void input\_month(int);

void output\_month();

void output\_month\_name();

int NextMonth(const Month &p);

};

**Month.cpp**

#include <iostream>

#include "Month.h"

using namespace std;

Month::Month(){

cout << "This is default Constructor.." << endl;

month = 0;

}

Month::Month(char\*a){

cout<<"This is parameterized constructor with month names.." <<endl;

map\_int(a);

}

Month::Month(int a){

cout<<"This is parameterized constructor with month as int.. " << endl;

month = a;

}

char\* Month::map\_char(int a){

char\* month\_name;

if(a == 1){

month\_name = "Jan";

}

else if(a == 2){

month\_name = "Feb";

}

else if(a == 3){

month\_name = "Mar";

}

else if(a == 4){

month\_name = "Apr";

}

else if(a == 5){

month\_name = "May";

}

else if(a == 6){

month\_name = "Jun";

}

else if(a == 7){

month\_name = "Jul";

}

else if(a == 8){

month\_name = "Aug";

}

else if(a == 9){

month\_name = "Sep";

}

else if(a == 10){

month\_name = "Oct";

}

else if(a == 11){

month\_name = "Nov";

}

else if(a == 12){

month\_name = "Dec";

}

return month\_name;

}

void Month::map\_int(char\* a){

if(a == "Jan"){

month = 1;

}

else if(a == "Feb"){

month = 2;

}

else if(a == "Mar"){

month = 3;

}

else if(a == "Apr"){

month = 4;

}

else if(a == "May"){

month = 5;

}

else if(a == "Jun"){

month = 6;

}

else if(a == "Jul"){

month = 7;

}

else if(a == "Aug"){

month = 8;

}

else if(a == "Sep"){

month = 9;

}

else if(a == "Oct"){

month = 10;

}

else if(a == "Nov"){

month = 11;

}

else if(a == "Dec"){

month = 12;

}

}

void Month::input\_month(int i){

month = i;

}

void Month::output\_month(){

cout<<"The month in integer is: "<< month << endl;

}

void Month::output\_month\_name(){

char\* month\_name = map\_char(month);

cout<<"The month name is: " << month\_name << endl;

}

int Month::NextMonth(const Month &p){

Month Temp;

Temp.month = (p.month + 1) % 12;

cout<<"This is next month function, the next month is: " << endl;

Temp.output\_month();

Temp.output\_month\_name();

return Temp.month;

}

int main(){

//Initializing constructors

Month M("Dec"), M1(5);

//Output Month and Names

M.output\_month();

M.output\_month\_name();

M1.output\_month();

M1.output\_month\_name();

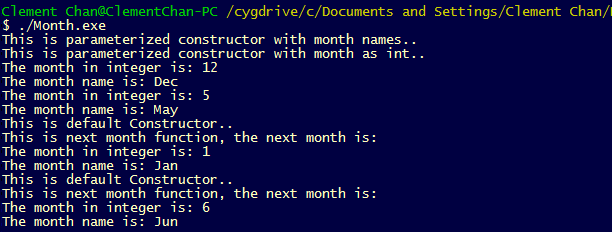
//Defining next Month

M.NextMonth(M);

M1.NextMonth(M1);

}

**Output**



**Question 2 – Dimes, Cents, Dollars, Thousands**

**Header File**

//Define the class that includes all the actions for counters

//Define the maximum constant that will be used for counting for the rest of the program

const int MAX\_COUNT = 9999;

class Counter{

private:

int count;

bool value;

int a;

int s;

int d;

int f;

public:

Counter();

void reset();

void incr1();

void incr10();

void incr100();

void incr1000();

//Define overflow actions

bool overflow();

//These functions limit the numbers between 1-9

void in\_cents();

void in\_dimes();

void in\_dollars();

void in\_10\_dollars();

//Display values

void Display();

void request\_overflow();

void enter(char);

};

**Count.cpp**

#include <iostream>

#include "Counter.h"

using namespace std;

//Defining method definitions

//Starting constructors

Counter::Counter(){

count = 0;

a = 0;

s = 0;

d = 0;

f = 0;

}

void Counter::reset(){

count = 0;

}

void Counter::incr1(){

count += 1;

}

void Counter::incr10(){

count += 10;

}

void Counter::incr100(){

count += 100;

}

void Counter::incr1000(){

count += 1000;

}

bool Counter::overflow(){

if (count > MAX\_COUNT){

value = true;

}

else{

value = false;

return value;

}

}

void Counter::in\_cents(){

a += 1;

incr1();

if(a > 9){

a = 1;

count -= 9;

}

}

void Counter::in\_dimes(){

s += 1;

incr10();

if(s > 9){

s = 1;

count -= 90;

}

}

void Counter::in\_dollars(){

d += 1;

incr100();

if(d > 9){

d = 1;

count -= 900;

}

}

void Counter::in\_10\_dollars(){

f += 1;

incr1000();

if(f > 9){

f = 1;

count -= 9000;

}

}

void Counter :: Display(){

if (count != 10000){

cout << "The Count is: " << count << endl;

cout << "The Amount of Cent is: " << a << endl;

cout << "The Amount of Dime is: " << s << endl;

cout << "The Amount of Dollar is: " << d << endl;

cout << "The Amount of 10 Dollars is: " << f << endl;

}

}

void Counter::request\_overflow(){

count = 10000;

value = overflow();

if(value == 1){

cout << "Count exceeds Maximum 9999" << endl;

exit(0);

}

}

void Counter::enter(char p){

if(p =='a'){

in\_cents();

}

else if(p == 's'){

in\_dimes();

}

else if(p == 'd'){

in\_dollars();

}

else if(p == 'f'){

in\_10\_dollars();

}

else if(p == 'o'){

request\_overflow();

}

else{

cout << "Please Re-Enter " << endl;

}

}

int main(){

char input;

Counter C;

cout<<"Please enter the digits (a = 1, s = 10, d = 100, f = 1000, o = overflow): "<<endl;

do{

cin >> input;

C.enter(input);

C.Display();

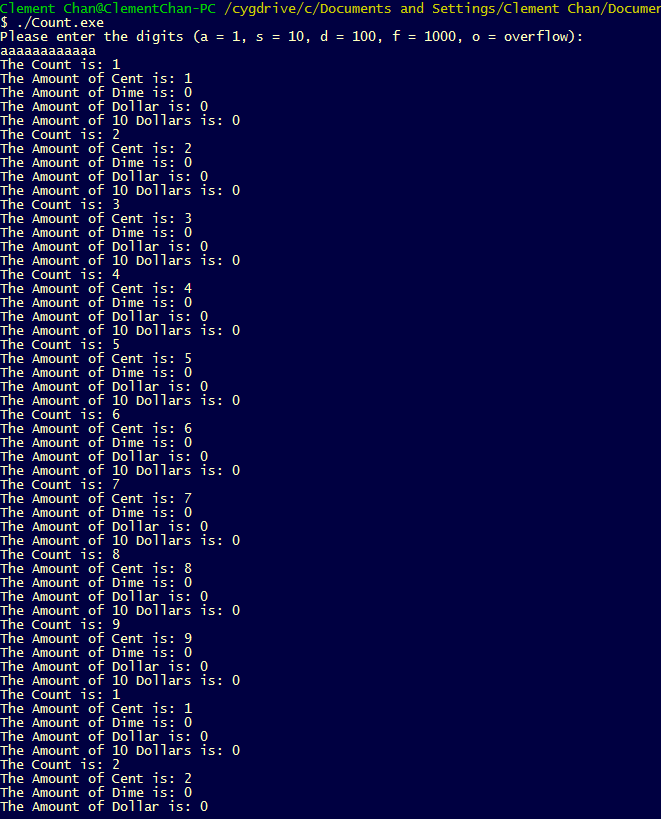
}while (input != 'o');

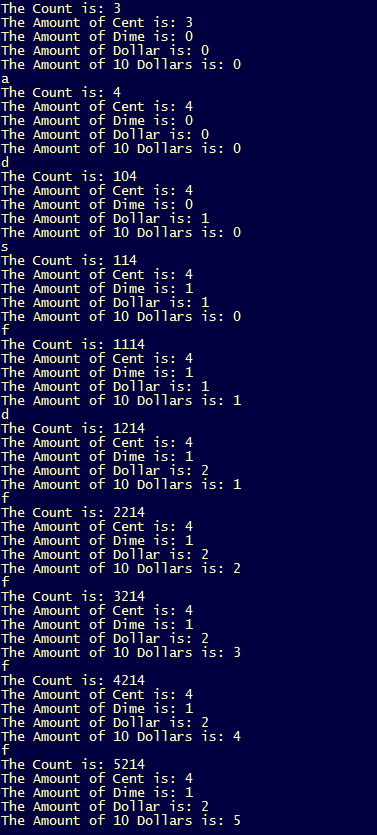
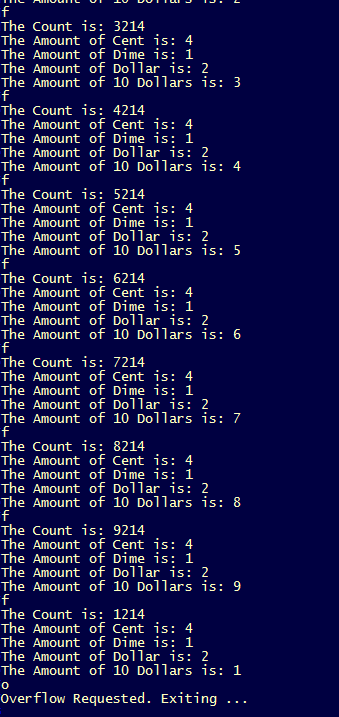
cout<<"Overflow Requested. Exiting ..." << endl;

return 0;

}

**Output**



**Question 3 – Hot Dog**

**Header File**

//Define the Hot Dog Stand Class for definition uses

class HotdogSt{

private:

int ID;

int num\_hd\_sold;

public:

static int total\_hd\_sold;

HotdogSt(int,int);

void JustSold();

void getStandSold();

int getID();

static int display\_total();

};

//Initializing the public static int

int HotdogSt::total\_hd\_sold = 0;

**Hot dog.cpp**

#include <iostream>

#include "Hotdog.h"

using namespace std;

//Method Definitions

HotdogSt::HotdogSt(int a, int b){

ID = a;

num\_hd\_sold = b;

total\_hd\_sold += b;

}

void HotdogSt::JustSold(){

cout << "Hotdog Stand " << ID << " just sold a hot dog !" << endl;

num\_hd\_sold += 1;

total\_hd\_sold += 1;

}

void HotdogSt::getStandSold(){

cout << "Hotdog Stand " << ID << " now sold " << num\_hd\_sold << " hot dogs" << endl;

}

int HotdogSt::getID(){

return ID;

}

int HotdogSt::display\_total(){

cout<<"The combined hot dog sold now is : ";

return total\_hd\_sold;

}

int main(){

HotdogSt HDT1(1,20), HDT2(2,30), HDT3(3,40);

cout<<HDT3.display\_total()<<endl;

HDT1.JustSold();

HDT2.JustSold();

HDT2.JustSold();

HDT2.JustSold();

HDT3.JustSold();

HDT1.getStandSold();

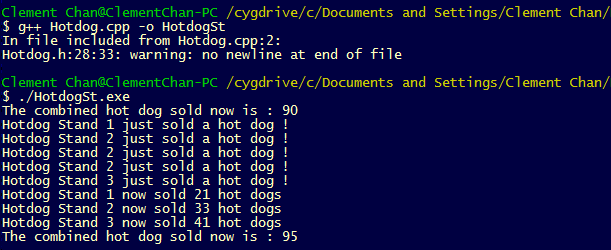
HDT2.getStandSold();

HDT3.getStandSold();

cout<<HDT1.display\_total()<<endl;

}

**Output**



**Question 4 – Histogram**

#include <iostream>

#include <vector>

using namespace std;

const int Max\_Size = 10;

int histogram[Max\_Size];

void init\_hist(){

for(int i = 0; i < Max\_Size ; i++){

histogram[i] = 0;

}

}

void Display\_Hist\_Table(){

int num\_of\_grades;

for(int i = 0; i < Max\_Size ; i++){

num\_of\_grades = histogram[i];

int a = 0;

if (i == 0){

cout<<"Score between " << i << " and " << i + 10 << " is|";

}

else{

cout<<"Score between " << i\*10 + 1 << " and " << (i+1)\*10 << " is|";

}

while (a < num\_of\_grades){

cout<< "\*";

a++;

}

cout<<" "<<endl;

}

}

int main(){

int input\_grades;

vector<double> v;

init\_hist();

cout<<"Please Enter the grade of the student from 0 to 100 points, -1 to complete entering: " <<endl;

do{

cin >> input\_grades;

v.push\_back(input\_grades);

if(input\_grades >= 0 && input\_grades <= 10){

histogram[0] += 1;

}

else if(input\_grades >= 11 && input\_grades <= 20){

histogram[1] += 1;

}

else if(input\_grades >= 21 && input\_grades <= 30){

histogram[2] += 1;

}

else if(input\_grades >= 31 && input\_grades <= 40){

histogram[3] += 1;

}

else if(input\_grades >= 41 && input\_grades <= 50){

histogram[4] += 1;

}

else if(input\_grades >= 51 && input\_grades <= 60){

histogram[5] += 1;

}

else if(input\_grades >= 61 && input\_grades <= 70){

histogram[6] += 1;

}

else if(input\_grades >= 71 && input\_grades <= 80){

histogram[7] += 1;

}

else if(input\_grades >= 81 && input\_grades <= 90){

histogram[8] += 1;

}

else if(input\_grades >= 91 && input\_grades <= 100){

histogram[9] += 1;

}

else{

cout<<"Input out of range" << endl;

cout<<"Exiting input ... " << endl;

}

}while(input\_grades > 0);

//Displaying the number of students in the class

cout<<"The size of the class with vector is: " << v.size() - 1 << endl;

//Displaying the max grade user entered

cout<<"The max grade entered for the class is: " << \*max\_element(v.begin(),v.end()) << endl;

//Displaying the statistics for the grades entered

for(int i = 0; i < Max\_Size; i++){

if (i == 0){

cout<<histogram[i]<<" students got between " << i << " and " << i + 10 << endl;

}

else{

cout<<histogram[i]<<" students got between " << i\*10 + 1 << " and " << (i+1)\*10 << endl;

}

}

cout<< " " << endl;

//Display the histogram

Display\_Hist\_Table();

return 0;

}

**Output**

