

Thank you, here is your change ..  
Twonnies = 0  
Loonies = 0  
Quarters = 0  
Dimes = 1  
Nickels = 0  
Enter how many dollars the item costs 3  
Enter how many cents the item costs 56  
Dollars = 3  
cents = 55  
Enter how many dollars the customer paid 4  
Enter how many cents the customer paid 00  
Thank you, here is your change ..  
Twonnies = 0  
Loonies = 0  
Quarters = 1  
Dimes = 2  
Nickels = 0  
Thank for for shopping at Keith's. The most expensive item you  
purchased cost:  
Dollars = 3  
cents = 55

**Hint:** The given incorrectly written program produces the following output:

```
ted@hvgray Midterm_Winter_2014 > bad_sum_of_first_five_negative
Please enter an integer number: 1
Please enter an integer number: 2
Please enter an integer number: 3
Please enter an integer number: 4
Please enter an integer number: 5
Please enter an integer number: 6
The sum of the first five negative numbers you have entered is: -4196160
```

**Another Hint:** Here is another sample output of the given incorrectly written program:

```
ted@hvgray Midterm_Winter_2014 > bad_sum_of_first_five_negative

Please enter an integer number: -1
Please enter an integer number: -1
Please enter an integer number: -1
Please enter an integer number: -1
Please enter an integer number: -1
Please enter an integer number: -1
The sum of the first five negative numbers you have entered is: -4196165
```

**Yet Another Hint:** The correctly written program produces the following output:

```
ted@hvgray Midterm_Winter_2014 > sum_of_first_five_negative
Please enter an integer number: 1
Please enter an integer number: -1
Please enter an integer number: 2
Please enter an integer number: 3
Please enter an integer number: -2
Please enter an integer number: -3
Please enter an integer number: -4
Please enter an integer number: 4
Please enter an integer number: -5
The sum of the first five negative numbers you have entered is: -15
```

**Question 3:** [60 points]

The Canadian coin system consists of the following coins:

- a two dollar coin commonly referred to as a 'twonnie' (worth 200 cents)
- a one dollar coin known as a 'loonie' (with value equal to 100 cents)
- a quarter (worth 25 cents)
- a dime (10 cents)
- a nickel (5 cents)

There used to be a coin called a penny (worth 1 cent) but it has been removed from circulation as of Feb. 12, 2013. Figure 1 shows the 5 coins in increasing order of monetary value (from nickel to twonnie):



(b) Consider the following program:

```
#include <iostream>
using namespace std;

int main()
{
    struct a_random_struct
    {
        int mick;
        float keith;
        char ron ;
    } ;

    a_random_struct v1, v2;

    v1.mick = 5 ;
    v1.keith = 10.0;
    v1.ron = 'r';

    v2.mick = 5 ;
    v2.keith = 10.0;
    v2.ron = 'r';

    if ( v1 == v2)
    {
        cout << "The two structs are the same" << endl;
    }
    else
    {
        cout << "The two structs are NOT the same" << endl;
    }

    return 0;
}
```

The output produced by the program is (choose only one of the following):

- (i) The two structs are the same
- (ii) The two structs are NOT the same
- (iii) Segmentation Fault
- ☒ (iv) No output is produced as the program does not compile
- (v) None of the above

The following struct called `coins` can be used to represent the amount of change given back to a customer (in terms of coins) :

```
struct coins
{
    int twonnie;
    int loonie;
    int quarter;
    int dime;
    int nickel;
};
```

Similarly, the following struct called `money` can be used to represent a monetary value in terms of dollars and cents ( 1 dollar = 100 cents):

```
struct money
{
    int dollars;
    int cents;
};
```

This question (**read the entire program before beginning to answer**) involves several user defined functions which receive arguments of the above type and perform operations on these arguments. The functions are used in a `main()` function as follows:

```
#include <iostream>
using namespace std;
```

```
struct money
{
    int dollars;
    int cents;
};
```

```
struct coins
{
    int twonnie;
    int loonie;
    int quarter;
    int dime;
    int nickel;
};
```

```
void how_much_does_that_thing_cost(floatmoney& cost)
```



```
f1(mr_keith);  
{  
    int mr_keith = 20 ;  
    f1(mr_keith);  
    f1::mr_keith;  
}  
f1(mr_keith);  
f2();  
cout << ::mr_keith << endl;  
return 0;  
}
```

**Question 2:** [20 points]

Consider the following program which compiles without any errors or warning messages:

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
  
    int number;  
    int sum ;  
    int how_many = 0;  
  
    while ( how_many < 6 )  
    {  
        cout << "Please enter an integer number: " ;  
        if ( number < 0 )  
            sum = sum + number;  
            how_many++;  
        cin >> number;  
    }  
  
    cout << "The sum of the first five negative numbers you have  
entered is: " << sum << endl;  
  
    return 0;  
}
```

The program, as it is written, contains **four** run-time errors which cause it to produce incorrect results. Identify these four errors and state how you would correct them so that the program will produce the correct output. The **correctly written program is to compute the sum of the first five negative integers entered from the keyboard.**

**Date:** Thursday, February 27, 2014

**Time Allowed:** 2 hours

**Material Allowed:** Course text book, course notes, ENCS approved electronic calculators, pencils, pens, eraser.

**Instructions:** Answer all 3 questions in the provided answer booklets. If you make any assumptions, clearly indicate so in your answer booklet.

**Question 1:** [20 points]

(a) Consider the following innocuous looking program:

```
#include <iostream>
using namespace std;

int main()
{

    int mick;
    unsigned long keith;

    keith = 4294967295; // this value was chosen for a
                      //specific reason...

    mick = keith;

    cout << "mick = " << mick << " keith = " << keith << endl;

    return 0;
}
```

The program produces the following output when it is run:

```
ted@deadflowers Programs > unsigned_2_int
mick = -1 keith = 4294967295
```

Explain in words why the value of variable mick is equal to -1 (instead of 4294967295).

**Hint:** ted@deadflowers Programs > g++ -o unsigned\_2\_int -pedantic -Wconversion unsigned\_2\_int.C  
unsigned\_2\_int.C: In function int main():  
unsigned\_2\_int.C:22:8: warning: conversion to int from long unsigned int may alter its value [-Wconversion]



```
{
// this function receives an argument named cost of
// type money and prompts the user to enter
// the dollars and cents it costs. The function then
// assigns these values which are entered from the keyboard
// to the argument cost which is to be modified by the function.

    cout << "Enter how many dollars the item costs " ;

// you complete the rest
}

void how_much_did_customer_give_to_pay(money& paid )
{

// this function receives an argument of type money called paid.
// The function prompt the user to enter the amount paid
// by the customer in dollars and cents and assigns these values
// to the argument called paid. The argument is to be modified
// by the function.

    cout << "Enter how many dollars the customer paid " ;
    cin >> money.dollars ;
    cout << "Enter how many cents the customer paid " ;
    cin >> money.cents ;
}

void round_up_down(money& cost)
{
// This function implements the rounding as per the rules
// of Table 1.
// The complete code of the function is given to you.
// You don't have to do anything for this function

if ( cost.cents % 5 != 0 ) // if not on a nickel multiple need to
                        // round up or down
{
    if ( cost.cents % 10 <= 2 )
    {
        cost.cents = cost.cents - (cost.cents % 10) ;
    }
    else
        if ( cost.cents % 10 <= 7)
        {
```

```
cost.cents = cost.cents - (cost.cents % 10) + 5;
}
else
{
cost.cents = cost.cents - ( cost.cents % 5 ) + 5 ;
}

// now adjust dollars fields if the cents field is >= 100
//after rounding up

if ( cost.cents >= 100 )
{
cost.dollars = cost.dollars + 1 ;
cost.cents = 0;
}
}

void print_the_money(money cost)
{
// This function receives by value an argument of type
// money called cost and prints out the values of the
// argument.

cout << "Dollars = " << cost.dollars << endl ;
cout << "cents = " << cost.cents << endl;
}

void print_the_change(coins change )
{
// This function receives by value an argument of type
// coins called change and prints out the values of
// the argument.

cout << "Twonnies = " << change.twonnies << endl;
cout << "Loonies = " << change.loonies << endl;
cout << "Quarters = " << change.quarters << endl;
cout << "Dimes = " << change.dimes << endl;
cout << "Nickels = " << change.nickels << endl;
}

bool did_customer_pay_enough(money cost, money paid)
{
// this function receives by value two arguments of type
// money representing the cost of an item and the amount
```



```

// paid for the item and returns true if the amount paid
// is greater or equal to the cost of the item and returns
// false otherwise.

// you give the rest of the code for this function.
}

coins compute_change(money cost, money paid)
{
    // This functions receives two arguments by value of type
    // money representing the cost of an item and the amount paid
    // (which will be greater than or equal to the cost) and computes
    // the required change using the minimum number of coins. The
    // function then returns a value of type coins as an answer.
    // Hint: Here is the starting point ..

    coins change;

    change.twonnie = 0;
    change.loonie = 0;
    change.quarter = 0;
    change.dime = 0;
    change.nickel = 0;

    int change_in_cents ;

    change_in_cents = ((paid.dollars*100) + paid.cents) - ((cost.dol-
    lars * 100) + cost.cents);

    // now determine the minimum number of twoonies, loonies,
    // quarterrs, dimes; and nickels required for this amount
    // of change_in_cents

    return change. ;
}

bool cost_greater_equal_to_max(money cost_of_item, money max)
{
    // This function receives two arguments of type money and returns
    // true if the cost_of_item is greater than or equal to the second
    // argument called max which holds the current value of the most
    // expensive item, otherwise it returns false.
    // you do the rest ...
}

```



Figure 1: Canadian coins

With the removal of the penny from circulation, vendors are required to either round up or round down the price of an item to the nearest nickel. For example, if the actual price of a coffee is \$1.86 the price charged to the customer is rounded down to \$1.85. A price of \$1.88 is rounded up to \$1.90. In general, a price of \$X.YZ is either rounded down or up according to the value of Z as given in Table 1:

Table 1: Rounding based on last digit of cents

| Actual Price | Rounded Price |
|--------------|---------------|
| \$X.Y0       | \$X.Y0        |
| \$X.Y1       | \$X.Y0        |
| \$X.Y2       | \$X.Y0        |
| \$X.Y3       | \$X.Y5        |
| \$X.Y4       | \$X.Y5        |
| \$X.Y5       | \$X.Y5        |
| \$X.Y6       | \$X.Y5        |
| \$X.Y7       | \$X.Y5        |
| \$X.Y8       | \$X.(Y+1)0    |
| \$X.Y9       | \$X.(Y+1)0    |



(c) Give the output produced by the following program:

```
#include <iostream>
using namespace std;

int main()
{
    int i, j, k ;
    float l;
    i = 5 ;
    j = 10;
    k = 7 ;

    l = (i++) * ( (--j) / k ) ;
    cout << i << " " << j << " " << k << " " << l << endl;

    return 0;
}
```

5 9 7 6

(d) Give the output produced by the following program:

```
#include <iostream>
using namespace std;

int mr_keith = 5 ;

void f1(int& mr_keith)
{
    cout << mr_keith << endl; 5
    cout << ::mr_keith << endl; 5
    mr_keith++;
}

void f2()
{
    int mr_keith = 10 ;
    cout << mr_keith << endl; 10
}

int main()
{
    int mr_keith = 15;
```

```
int main()
{
    money cost_of_item;
    money most_expensive_item;
    money amount_paid;
    coins change;

    int number_of_items_purchased = 0;
    most_expensive_item.dollars = 0;
    most_expensive_item.cents = 0;

    while(number_of_items_purchased < 2)
    {
        how_much_does_that_thing_cost(cost_of_item);

        number_of_items_purchased++;

        round_up_down(cost_of_item);

        print_the_money(cost_of_item);

        how_much_did_customer_give_to_pay(amount_paid);

        if ( did_customer_pay_enough( cost_of_item, amount_paid ) )
        {
            cout << "Thank you, here is your change .. " << endl;
            change = compute_change(cost_of_item, amount_paid);
            print_the_change(change);
        }
        else
        {
            cout << "I'm sorry but the item costs more than the amount you
            have provided"
            << endl;
        }

        if ( cost_greater_equal_to_max(cost_of_item, most_expensive_item)
        )
        {
            most_expensive_item.dollars = cost_of_item.dollars;
            most_expensive_item.cents = cost_of_item.cents;
        }
    }
}
```