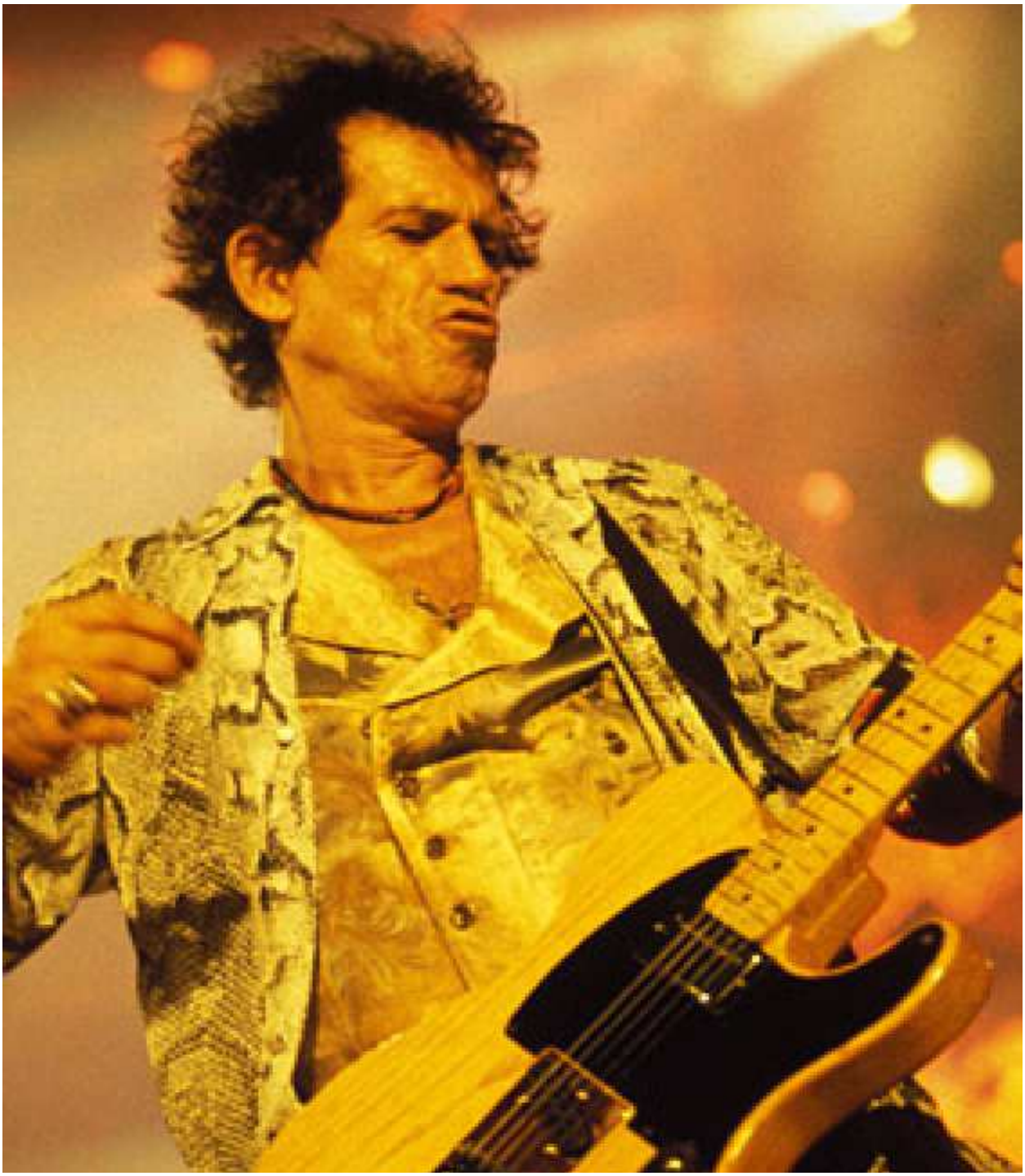
CONCORDIA UNIVERSITY
Department of Electrical and Computer Engineering
COEN 243/4 Section S : Programming Methodology I Winter 2003
Midterm Exam I

"I can't get no C++."



Instructor: Tadeusz Obuchowicz
Date: Feb. 13, 2003
Time: 11:45 A.M - 1:00 P.M

Materials Allowed: Pencils, pens, rulers, erasers, electronic calculators.

Instructions: **Attempt** all 4questions. If you make any assumptions, indicate so in your booklet.

Question 1: [25 points]

One may define the **integer logarithm base y** of some positive integer **x** as the **smallest value** of **a** such that:

$\mathbf{y}^{\mathbf{a}} >= \mathbf{x}$.

In other words, we say that $\mathbf{a} = \log_{\mathbf{y}}(\mathbf{x})$ if $\mathbf{y}^{\mathbf{a}} >= \mathbf{x}$.

For example, the $\log_3$ of 25 is equal to 3 since $3^3 = 27 >= 25$. The value of $\mathbf{y}^{\mathbf{a}}$ may be computed as simply the product **y** * **y** * **y** * **y** ... * **y** (in other words **y** multiplied by itself a total of **a** times.

Write a C++ program which prompts the user to input two integer values corresponding to **x** (the value whose logarithm we are to compute) and **y** (the base which is to be used in computing the logarithm). The program then computes the answer (the value of **a**) and displays the result. Typical interaction with the program would be as:

ted@townshend Programs 7:10pm >log_general_base
Please enter the value x you wish to compute the integer log of 8
Please enter the base of the logarithm 2
The answer is : 3

ted@townshend Programs 7:32pm >log_general_base
Please enter the value x you wish to compute the integer log of 345
Please enter the base of the logarithm 5
The answer is : 4

ted@townshend Programs 7:33pm >log_general_base
Please enter the value x you wish to compute the integer log of 48
Please enter the base of the logarithm 4
The answer is : 3

Do not use any of the math library functions found in the <cmath> library (or any other library). Your program should be general purpose and work for any positive integer x and any positive base y. Your program should make use of only the built-in arithmetic operators of addition

and multiplication in addition to any other control structures you deem necessary to solve the problem (i.e. looping constructs might be helpful...)

Question 2: [25 points]

Write a C++ function called is_even which receives an integer value as an argument and returns the boolean value of true if the passed argument is an even number, otherwise it returns false. Write a second C++ function called is_odd which receives an integer value as an argument and returns the boolean value of true if the passed argument is an odd nmber, otherwise it returns false. Use the following prototypes:

bool is_even(int keith);
bool is_odd(int mick);

Use these two functions within a main program to find the sum of all the even integers, the sum of all the odd integers, as well as the sum of all the integers which lie in between the range 0 and **n** inclusive, where **n** is a user specified input. The program should check that the sum of the evens added together with the sum of the odds is equal to the sum of all the integers.

For those of you who are interested in numerology, here is the output:

ted@townshend Programs 8:47pm >even_odd_functions2
Please enter an ending value 1000
The sum of the evens is 250500
The sum of the odds is 250000
The sum of all is 500500
The sum of evens and odds is 500500

Question 3 : [25 points]

What will be the output produced by the following program?

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

int out_of_scope = 5;

void keith()
{
 static  int out_of_scope = 3;
 cout << ++out_of_scope << endl;
 cout << ::out_of_scope << endl;
}

int main()
```

```
{
int out_of_scope = 100;
{
int out_of_scope = -100;
cout << "Value of out_of_scope is " << out_of_scope << endl;
for(int mick = 0; mick < 2 ; mick++)
  keith();
}
::out_of_scope = ::out_of_scope + 1;
cout << ::out_of_scope << endl;
cout << out_of_scope << endl;
return 0;
}
```

Question 4: [25 points]

Write a C++ function with the following prototype:

void multiply_fractions(int num1, int denom1, int num2, int denom2,
int& reduced_num, int& reduced_denom)

This function receives 4 value arguments which represent the numerators and denominators of two fractions, the function performs the multiplication of these two fractions and REDUCES the resulting product fraction to its LOWEST form. The function then uses the two reference arguments to return to the calling program the resulting product fraction. In order to perform this task, the function multiply_fractions makes use of a function called greatest_common_divisor which returns the greatest common divisor of its two passed integer arguments. The prototype for greatest_common_divisor is:

int greatest_common_divisor(int num, int denom)

The overall structure of the program is thus:

```
#include <iostream>
int greatest_common_divisor(int num, int denom);
void multiply_fractions(int num1, int denom1, int num2, int   denom2, int&
reduced_num, int& reduced_denom);

int main()
{
int a,b,c,d;
int mick, keith;
cout << "Please enter 4 integers ";
cin >> a >> b >> c >> d;
multiply_fractions(a,b,c,d, mick, keith);
cout << "The reduced fraction is " << mick << "/" << keith << endl;
return 0;
}
```