PASS MOCK EXAM – *FOR PRACTICE ONLY*

Course: ECOR 1606 Facilitator: Dane Levere

Dates and locations of take-up: Wednesday April 23rd, 2014 12:00pm-3:00pm LA C164

IMPORTANT:

It is **most beneficial** to you to write this mock midterm **UNDER EXAM CONDITIONS**. This means:

- Complete the midterm in 3 hour(s).
- Work on your own.
- Keep your notes and textbook closed.
- Attempt every question.

After the time limit, go back over your work with a different colour or on a separate piece of paper and try to do the questions you are unsure of. Record your ideas in the margins to remind yourself of what you were thinking when you take it up at PASS.

The purpose of this mock exam is to give you practice answering questions in a timed setting and to help you to gauge which aspects of the course content you know well and which are in need of further development and review. Use this mock exam as a ***learning tool*** in preparing for the actual exam.

Please note:

- Come to the PASS session with your mock exam complete. There, you can work with other students to review your work.
- Often, there is not enough time to review the entire exam in the PASS session. Decide which questions you most want to review – the Facilitator may ask students to vote on which questions they want to discuss.
- Facilitators do not bring copies of the mock exam to the session. Please print out and complete the exam before you attend.
- **Facilitators do not produce or distribute an answer key for mock exams.** Facilitators help students to work together to compare and assess the answers they have. If you are not able to attend the PASS session, you can work alone or with others in the class.

Good Luck writing the Mock Midterm!!

Question 1

Given the following function:

```
int execute(int x[], int &y, int z){
for (int i = z; i >= 0; i--){
        x[i] = y++;
        }
        z += 1;
        return z - 2;

}
```

a) What will be the output if the following code is executed?

```
int b = 0, c = 4, d;
int a[5] = {0};
d = execute(a, b, c);
for (int i = 0; i <= c; i++){
        cout <<  a[i] << " ";
}
cout << endl;
```

b) What will be the output if the following code is executed?

```
int b = 2, c = 3, d;
int a[5] = {0};
d = execute(a, b, c);
c *= 0.8;
cout<<b%4 <<""<<c<<""<<d*1.25<<endl;
```

c) What will be the output if the following code is executed?

```
int b = 9, c = 2, d;
int a[5] = {0};
if ((b % c == 1) || ((a[1] + 3 * 2 % 2 == 0) && (a[c] / c == b + 2 * a[b]))){
        b = 2;
        c = 1;
} else {
        b = 1; c = 2;

}
d = q1(a, b, c);
cout << b << " " << c << " " << d << endl;
```

d) What will be the output if the following code is executed?

```
double a = 1./3;
for (int i = 0; i <= 6; i+=2){
        cout << setfill('X') << setprecision(i) << setw(i+2) << ++a << endl;

}
```

Question 2

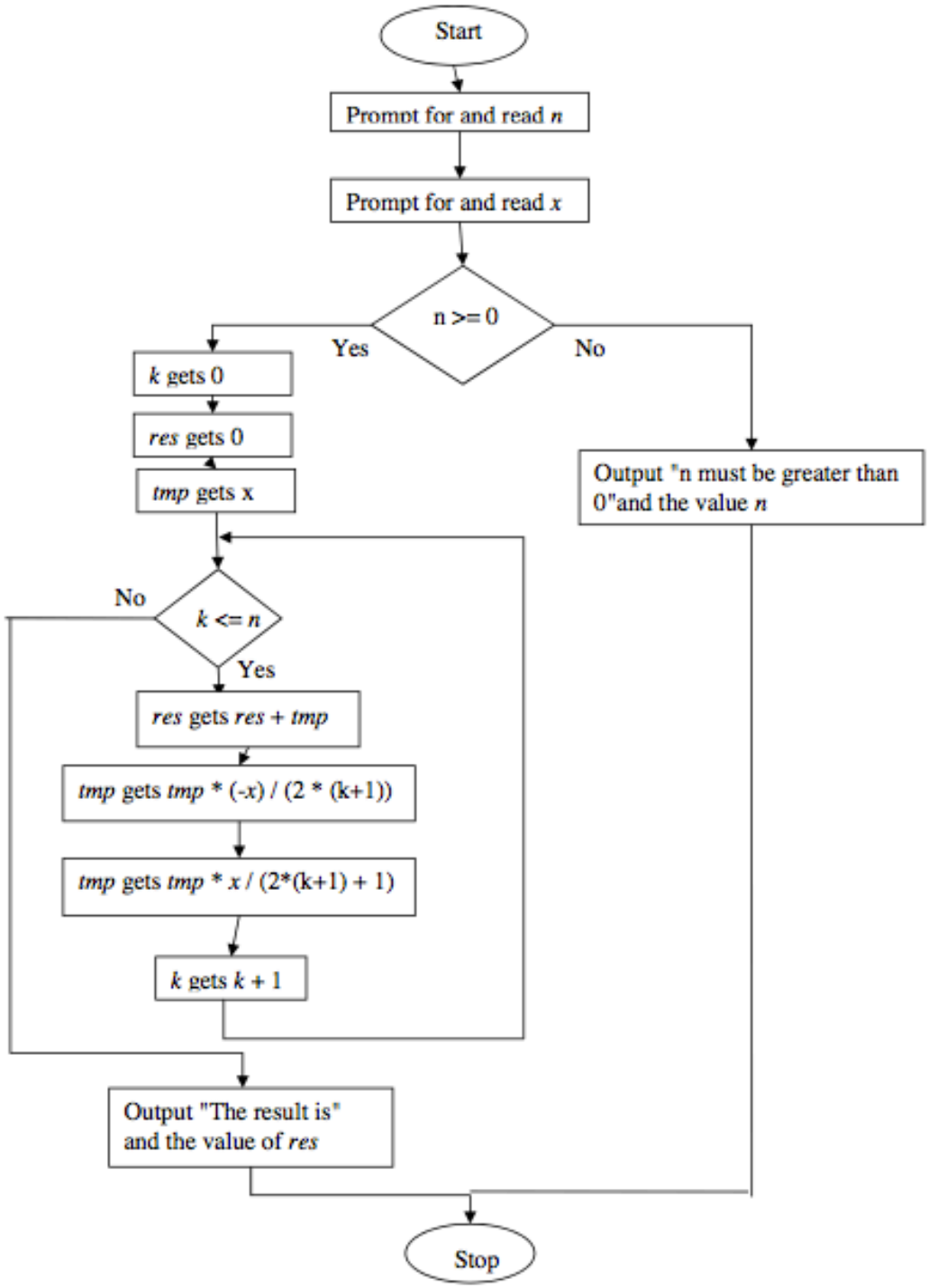
Write a function that accepts an array of integer values and the number of values in this array. It should return true if any of the values in the array is the square of one of the **other** values, and false otherwise

Note the "**other** values". Having a "1" somewhere in the array does NOT guarantee a true result. Your function must be consistent with the sample call below:

```
int values[20];
.....
if (checkSquares (values, 20)) {
.... // one of the values is the square of one of the other values }
```

Question 3

This question involves converting a flowchart into the corresponding C++ code. You are NOT required to produce the program framework or to declare any variables. Simply translate the logic represented by the flowcharts into C++ code.



Question 4

State what is wrong with the following snippets of code (assume all variables and file streams are setup and initialized properly where applicable). Fix the errors, re- writing the code if necessary.

a) calculates the sum of numbers stored in a file

```
sum = 0;
while (!fin.eof() | !fin.fail()){
    fin >> input;        sum = sum + input;
}
cout << sum << endl;
```

b) Opens file located at: files\data01\data040613.txt and checks for errors

```
ifstream fin;
fin.open("files\data01\data040613.txt");
if (fin.fail()){
    cout << "FILE ERROR: Unexpected Input\n";
    fin.ignore(MAX_INT,'\n'); fin.clear();
}
```

c) Asks user for filename and checks for errors

```
cout << "Enter a filename: ";
cin >> filename;
while (cin.fail()){
        cout << "Error opening file\n";
        cin.clear();
        cout << "Enter a filename: ";
        cin >> filename;
}
fin.open(filename);
```

Question 5

File "data.txt" is supposed to contain between 10 and 100 integer values (inclusive of these limits) and all of the values are supposed to be different (i.e. there should not be any duplicate values). Write a program that reads this file and verifies that it is valid. Your program should terminate after having output exactly one of the following messages:

. "File cannot be opened"

. "File contains bad data"

. "File contains too many values"

. "File contains too few values"

. "File contains at least one duplicate value"

. "File is OK"

ECOR 1606 Final Lab Test Crib Sheet and Hints (Note: 2 pages long)

CONTROL STRUCTURES

simple

if:

```
if (boolean exp) {
   statements // body
}
```

if-then-else:

```
if (boolean exp) {
   statements // true part
} else {
   statements // false part
}
```

multi-way if:

```
if (boolean exp) {
   statements // part 1
} else if (boolean exp) {
   statements // part 2
} else if (boolean exp) {
   statements // part 3
… // and so on
} else { // an else part is
            optional
   statements // else part
}
```

while loop (pre-test):

```
while (boolean exp) {
   statements // body
}
```

do-while loop (post-test):

```
do {
   statements // body
} while (boolean exp);
```

for loop:

```
for (exp1; exp2; exp3) {
   statements // body
}
```

is equivalent to:

```
exp1;
while (exp2) {
   same statements
   exp3;
}
```

break statement:

```
break;
```

– causes an exit from the enclosing loop.

continue statement:

```
continue;
```

– sends control back to the top of the enclosing loop.

CALL BY …

```
int sample (int a, int &b);
```

"a" is call-by-value (just like a regular variable but given a value when the function is called).
"b" is call-by-reference. all operations on "b" actually operate on the variable supplied.

MODEL PROGRAM

```
#include <iostream>
#include <cmath>
#include <iomanip>

using namespace std;

int add(int x, int y) {
    int result;
    result = x + y;
    return result;
}

int  main() {
    int a, b, c;
    cout << "Enter two values: ";
    cin >> a >> b;
    c = add(a, b);
    cout << "The answer is " << c << endl;
    system("PAUSE");
    return 0;
}
```

EXPRESSIONS

<	is less than
>	is greater than
<=	is less than or equal to
>=	is greater than or equal to
==	is equal to
!=	is not equal to
||	OR (either side is true)
&&	AND (both sides are true)
!	NOT (changes true to false, false to true)
%	modulus (gives remainder from division)

X++ means "use value of X, then increment X"
X-- means "use value of X, then decrement X"
++X means "increment X, then use new value"
--X means "decrement X, then use new value"

X += Y is equivalent to X = X + (Y)
(same idea for -=, *=. and /=)

OUTPUT (use iostream, iomanip)

```
cout << setiosflags(ios::fixed | ios::showpoint);  // forces use of non-scientific notation and the display of zeroes
                                                   // to the right of the decimal point. this remains in effect until changed.
cout << setprecision(value);       // selects the number of digits to be displayed to the right of the decimal point
                                   // when outputting double values.  the choice remains in effect until changed.
cout << setw (value) << …          // indicates that the next value output is to occupy the specified number of
                                   // columns.  a one shot deal – affects only the next value output
cout << setfill(ch);           // selects the fill character to be used when padding output values to a specified width.
                               // the choice remains in effect until changed.
```

LIBRARY FUNCTIONS

```
double fabs(double x);   returns the absolute value of "x", for real numbers
int abs(int x);      returns the absolute value of "x", for integers
double log (double x)    natural log (log base e)
double log10(double x)  log base 10
double exp(double x)     returns "e" to power of  "x"
double sqrt(double x)    returns the square root of "x"
double pow (double x, double y);          returns  "x" to the power of "y"
double sin(double x);    returns the sine of "x" (note:  "x" is in radians)
double cos(double x);    returns the cosine of "x" (note:  "x" is in radians)
double asin(double x);   returns the inverse sin of "x" (in radians)
double acos(double x);   returns the inverse cosine of "x" (in radians)
double sinh(double x);   hyperbolic sin
double cosh(double x);   hyperbolic cosine
```

Seven Deadly Lab Final Test Sins

Stuck? Check that you aren't making any of the following mistakes:

1/. Integer division.

```
(1 / 5) is zero!
```

2/. Assignments in conditions:

```
if (a = b) {  // a disaster
```

3/. Legal (but nonsensical) logical expressions:

```
if  (7 < a < 67) {
if ((a || b || c) == 0) {
```

4/. Variables with the same name as a function.

```
sin = sin(x);
```

error: 'sin' cannot be used as a function

5/. Missing multiplication operators in expressions.

```
a = b(4 + 7);
```

error: 'b' cannot be used as a function

6/. Variables that are not properly initialized.

7/. Improperly chosen functions.

do you have *abs* where you want *fabs*?

do you have *log* and where you need *log10* (or vice versa)?