Test 1: CPS 08

Owen Astrachan

October 7, 1994

Name: ______________________________________

Honor code acknowledgement (signature) ____________________________

<table>
<thead>
<tr>
<th>Problem</th>
<th>value</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1</td>
<td>8 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 2</td>
<td>8 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 3</td>
<td>8 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 4</td>
<td>10 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 5</td>
<td>10 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 6</td>
<td>10 pts.</td>
<td></td>
</tr>
<tr>
<td>TOTAL:</td>
<td>54 pts.</td>
<td></td>
</tr>
</tbody>
</table>
PROBLEM 1:  (Vocabulary: 8 points)

For each of the words/phrases below, circle the definition that is the best description as it pertains in the context of computer science, programming, and C/C++.

1. **Makefile**
   
   (a) a file that must be compiled ahead of all other files before linking can occur.
   
   (b) a file used in compiling programs that insulates the user from the need to know where certain files exist and how the compiler is invoked.
   
   (c) a Unix command that creates a file without the need to use emacs or some other editor.

2. **constructor**
   
   (a) A method of building programs by implementing the programs one piece at a time, ensuring that all pieces work before proceeding to the next piece.
   
   (b) A type that is part of the standard C or C++ language (e.g., int or double) as opposed to one defined by the user.
   
   (c) A function that is automatically invoked and that initializes data fields of a class.

3. **emacs**
   
   (a) An editor and environment in which to develop programs.
   
   (b) A compiler for C++ programs.
   
   (c) A debugger used with C++ programs.

4. **short-circuit evaluation**
   
   (a) A term used to express the precedence hierarchy of operators, e.g., that the operator * has higher precedence than +.
   
   (b) A term used to describe how evaluation of the logical operators && and || works in evaluating their operands from left-to-right.
   
   (c) A term used to describe the mechanism by which parameters are passed by reference (e.g., when a parameter is defined as int & small).

PROBLEM 2:  (Header files: 8 points)

For each function, type, or class on the left, list the header file needed for its use. All header files are not used and a header file can be listed more than once. If no header file is needed (e.g., the type is built-in) put 0.

<table>
<thead>
<tr>
<th>Function/Type</th>
<th>Header File</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>0. no header file needed</td>
</tr>
<tr>
<td>Dice</td>
<td>1. iostream.h</td>
</tr>
<tr>
<td>String</td>
<td>2. math.h</td>
</tr>
<tr>
<td>ifstream</td>
<td>3. CPstring.h</td>
</tr>
<tr>
<td>cout</td>
<td>4. balloon.h</td>
</tr>
<tr>
<td>double</td>
<td>5. fstream.h</td>
</tr>
<tr>
<td>Balloon</td>
<td>6. dice.h</td>
</tr>
<tr>
<td>sqrt</td>
<td>7. stdio.h</td>
</tr>
</tbody>
</table>
**PROBLEM 3:**  \((\text{Trace/Debug: 8 points})\)

The program below prints three lines of output with two numbers on each line.

```cpp
#include <iostream.h>

int Test(int first, int second)
// postcondition: returns some value
{
    int a = first + second;
    if (first == second)
    {
        return a;
    }
    else if (first > second)
    {
        return 2*a;
    }
    else
    {
        return a + 5;
    }
}
main()
{
    int a,b,c;
    a = 3; b = 7; c = 15;
    cout << a + c/b << " " << a + c % b << endl;
    cout << (c == 5*a) << " " << (3*b + 1.5)/c << endl;
    cout << Test(b,c) << " " << Test(c,b) << endl;
}
```

What is the output of the program?

If the line `if (first == second)` is replaced with `if (first = second)` in the function `Test` the third line of output will change. What is the new third line of output?
PROBLEM 4: (Average: 10 points)

You are to complete a code segment that will permit the user to enter integers until the number 0 is entered. After all the numbers are entered, the average of the “special” numbers should be printed. A number is special if the function IsSpecial, whose header is given below, returns 1.

```cpp
int IsSpecial(int num)
// precondition: num > 0
// postcondition: returns 1 if num is special, else returns 0
```

For example, in the input sequence below only the numbers 5 and 8 are special (i.e., IsSpecial(5) == 1, IsSpecial(8) == 1, but IsSpecial(4) == 0).

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>enter a number&gt; -3</td>
</tr>
<tr>
<td>enter a number&gt; 4</td>
</tr>
<tr>
<td>enter a number&gt; 5</td>
</tr>
<tr>
<td>enter a number&gt; -7</td>
</tr>
<tr>
<td>enter a number&gt; 2</td>
</tr>
<tr>
<td>enter a number&gt; -1</td>
</tr>
<tr>
<td>enter a number&gt; 8</td>
</tr>
<tr>
<td>enter a number&gt; 0</td>
</tr>
<tr>
<td>average of special numbers = 6.5</td>
</tr>
</tbody>
</table>

The code segment is partially given below:

```cpp
int total = 0;
int numSpecial = 0;
int num;

while ( ) {
    // code here
}

cout << "average of special numbers = " << avg << endl;
```


PROBLEM 5: (Surface Area: 10 points)

Doug Cooper says that the surface area (in square meters) of a person can be approximated by the formula

\[
surface\ area = 7.184^{-3} \times \text{Weight}^{0.452} \times \text{Height}^{0.725}
\]

where weight is measured in kilograms and height in centimeters. The program below is designed to allow a user to calculate a person's surface area. You are to fill in the main function so that the program would run and generate output like that shown below.

No code should be added except in main.

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

void GetVitals(double & weight, int & height)
// postcondition: sets weight (kgs) and height (cms)
{
    cout << "enter weight (in lbs.): ";
    cin >> weight;
    weight /= 2.2; // convert to kilograms

    cout << "enter height (in inches): ";
    cin >> height;
    height *= 2.54; // convert to centimeters
}

double SurfaceArea(double height, double weight)
// precondition: height and weight given in cms and kgs respectively
// postcondition: returns surface area of person with given height/weight
{
    return 0.007184 * Pow(weight,0.452) * Pow(height,0.725);
}

main()
{
    output
    enter weight (in lbs.): 155
    enter height (in inches): 73
    your surface area = 2.6119 sq. meters
}
```
**PROBLEM 6**: *(Elementary, my dear Watson: 10 points)*

A class `Element` for manipulating elements from the periodic table of elements is described below. For this problem assume that there are 103 elements with consecutive atomic numbers from 1 to 103. The header file for the class is "elements.h".

```cpp
// class for manipulating elements (as in periodic table)

Element(int atomNumber) // create element with given atomic number
    e.g., Element hydrogen(1); creates hydrogen (atom. number = 1)
    Element carbon(6); creates carbon (atom. number = 6)

String Name(); // returns name of element,
    e.g., "hydrogen", "carbon" for hydrogen and carbon respectively

String Symbol(); // returns element's symbol,
    e.g., "H" for hydrogen, "C" for carbon

double Weight(); // returns atomic weight of element,
    e.g., 1.008 and 12.011 for hydrogen and carbon respectively

int AtomicNumber(); // returns atomic number of element,
    e.g., 1 for hydrogen and 6 for carbon

void SetAtomic(int num); // change atomic number to num
    e.g., elem.SetAtomic(8); sets elem to represent oxygen
    (the 8th element)
```

```cpp
class Element{
public:
    Element(int atomNumber); // element will have atomic # = parameter
    String Name(); // returns name of element
    String Symbol(); // returns symbol of element
    double Weight(); // returns weight of element
    int AtomicNumber(); // returns atomic number of element
    void SetAtomic(int num); // change atomic number to num

private:
    // stuff here that you can't see
};
```

**Part A:**
The atomic number of iron is 26. Complete the statements below so that the symbol of the element iron and its atomic weight are printed in some format (you must use member functions to provide the values of the symbol and weight).

```cpp
Element
cout <<
```
Part B:
Write a program that permits the user to enter the name of an element and that prints the atomic number,
and weight of that element. The user should be allowed to enter names until control-D is entered to signify
no more data (end-of-file).

```cpp
#include <iostream.h>
#include "elements.h"

main()
{

    String name;
    cout << "enter name of element (control-D to finish): ";

    while (cin >> name){

        cout << "enter name of element (control-D to finish): ";
    }
}
```

(output)

```
enter name of element (control-D to finish): copper
copper: number = 29, weight = 63.54
enter name of element (control-D to finish): kryptonite
kryptonite: not a recognized element
enter name of element (control-D to finish): phosphorus
phosphorus: number = 15, weight = 30.975
enter name of element (control-D to finish): ^D
```

(over for extra credit)
The formula below yields an approximation of $\pi/4$:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$

The more terms of the formula used, the better the approximation (up to a point). Write a function `PiApprox` that uses the formula above to approximate $\pi$. The value of the parameter `numTerms` provides the number of terms of the formula to use. For example, if one term is used then $\pi/4 = 1$ so $\pi = 4$. If three terms are used the $\pi/4 = 1 - 1/3 + 1/5$ so $\pi = 3.468$ since $1 - 1/3 + 1/5 = 0.867$.

(5 points)

double PiApprox(int numTerms)

// postcondition: returns approximation to pi using numTerms terms
{
