

Dr. Rodger

String class

```

int length();                // returns length of string
string substr(int pos, int len); // returns string of len chars
                                // starting at position pos
string Downcase();          // returns lower case of string
string Upcase();            // returns upper case of string
bool Contains(string s);    // returns true if string contains s
int find(string s);         // position at which s begins, or NPOS

```

Dice class

```

class Dice{
public:
    Dice(int sides); // constructor, sides specifies number of sides
    int Roll();      // return the random roll between 1 and num. of sides
    int NumSides();  // returns how many sides this die has
    int NumRolls();  // returns num. of times this die rolled
private:
    // not shown
};

```

PROBLEM 1 : (Output: (16 pts))

Give the output of the following statements.

Show blanks by putting an "_" for each blank printed.

```

int num = 5;
int score = 22;
string name = "Ada Lovelace";
string word = "fish";
double x = 2.0;

cout << score/num << " " << score % num << endl;
cout << "\num/x\" << " " << num/x << endl;
cout << 3678 << endl << " " << score + 3 + num * 2 << endl;
cout << "sword" + word << " " << word << endl;
cout << name.find("o") << " " << word.length() << endl;
cout << name.Upcase().substr(1,2) << endl;

if (num > 8)
    cout << "one" << endl;
else if (num > 3)
    cout << "two" << endl;
else if (num > 0)
    cout << "three" << endl;

```

```

int k;
for (k = -2; k < 10; k+=4)
{
    cout << k << " ";
}
cout << endl;

```

Output

PROBLEM 2 : (*Get Rid of Junk*: (12 pts))

Every Saturday there are many ads for yard sales, with similar wording in most of the ads.

Write a function named *YardSale* that is given the type of sale, the number of items for sale, and the address of the sale, and prints an ad for the paper. If the number of items for sale is less than 50, the actual number is not printed, since a low number may not entice people to come to the sale.

The two function calls below should generate the output shown.

```

YardSale("Baby Furniture", 10, "103 East Main Street.");
cout << endl;
YardSale("Misc", 120, "end of West Outer Drive.");

```

Corresponding Output

Baby Furniture yard sale. Lots of items. You can start early at 8am Saturday. Location is 103 East Main Street.

Misc yard sale. 120 items. You can start early at 8am Saturday. Location is end of West Outer Drive.

Write the function *YardSale* below.

PROBLEM 3 : (*The Other Half of a Roll*: (12 pts))

Write the function *MatchWithNum* whose header is given below. *MatchWithNum* takes a parameter *num* and continues to roll a pair of 6-sided dice until one of the dice is equal to *num*. The value of the other dice is then returned. The Dice class is shown on page 2.

Here is an example where *four* rolls were necessary for the call *MatchWithNum*(5). The first roll gives 4 and 1, the second roll gives 3 and 2, the third roll gives 2 and 2, and the fourth roll gives 5 and 2. *MatchWithNum* would return 2 in this case (the value of the other dice in the roll of 5 and 2). In another example, for the call *MatchWithNum*(4), the first roll is 4 and 4. Only one roll is necessary, and *MatchWithNum* would return 4 in this case.

Complete *MatchWithNum* below the following header.

```

int MatchWithNum(int num)
// precondition: 1 <= num <= 6
{

```

```
}
```

PROBLEM 4 : (*What do I Call you?* : (12 pts))

Write the function *RearrangeName* whose header is given below. *RearrangeName* takes a name in the form *firstname middlename lastname* (3 words, words separated by a blank) and an integer with value 1 if the firstname is preferred and value 2 if the middlename is preferred. *RearrangeName* returns a string formed with the lastname first followed by a comma, followed by the firstname and middlename. The name that is preferred is put in parenthesis.

Here are examples for two calls, showing their return value.

<u>Call</u>	<u>value returned</u>
<code>RearrangeName("Nancy Ann Kerkhoff", 2)</code>	<code>"Kerkhoff, Nancy (Ann)"</code>
<code>RearrangeName("Fred Sam Ritchie", 1)</code>	<code>"Ritchie, (Fred) Sam"</code>

Complete *RearrangeName* below the following header. (see the string class on page 2.)

```
string RearrangeName(string name, int nick)
// precondition: name has 3 words, words separated by a blank. nick is 1 or 2
{
}
}
```

PROBLEM 5 : (*The Aces Book:* (12 pts))

A class *DukeSchedule* is designed for being able to get information about a course for a particular semester. One can obtain the professor's name for a course, or the days and time the course meets.

Below is an example usage of this class.

```
DukeSchedule student(1998, "spring"); // spring 1998 semester
cout << student.CourseToProf("CPS", 6) << endl; // print prof for CPS 6
cout << student.CourseToTime("CPS", 6) << endl; // print and days time for CPS 6
```

The output corresponding to this example is below.

```
Rodger
TH 2:15
```

Here is the definition of the *DukeSchedule* class.

```

class DukeSchedule
{
    public:
        DukeSchedule(int year, string semester); // constructor, semester is
                                                // one of spring, fall, or summer
        string CourseToProf(string course, int number);
            // returns the name of the professor teaching the course
        string CourseToTime(string course, int number);
            // returns the time and days the course is taught
    private:
        // not shown
};

```

Write the function *PrintAllCourses* whose header is given below. *PrintAllCourses* does the following:

- Asks the user to type in the name of a professor (one word)
- Prints info about all the CPS courses this professor taught in the **fall 1996 semester**. In particular, the course number, and the days and time this course meets is printed. (CPS course numbers run from 1 to 399.)

For example, if the user typed the following input:

Rodger

Then the output of `PrintAllCourses()` would be (Prof. Rodger taught three courses in fall 1996, CPS 6, CPS 149 and CPS 100):

```

Enter Professor's name: Rodger
CPS 6 meets MW 2:30
CPS 100 meets TH 10:55
CPS 149 meets H 5:30

```

Complete *PrintAllCourses* below.

```

void PrintAllCourses()
{

}

```