NAME _____

Final Exam, 3/20/2018
CIS330, Winter 2018

You have 2 hours to complete this exam. You can use one letter-sized sheet of notes. Feel free to separate pages -- if you do, make sure to write your name on each sheet. Write your answers carefully and legibly. Partial answers are better than no answer. Feel free to skip around and go back to an earlier question later. When you are asked to write a program, writing comments is optional but you are expected to use good style and indent appropriately to make your code easier to follow. Write down assumptions you are making.

A perfect score is **100 points** on this exam out of 145 available points. If you earn a score greater than 100, the extra points will be applied to your midterm score up to 100 total for the midterm.

Good luck!

PROBLEM	POSSIBLE	SCORE
1	12	
2	30	
3	45	
4	35	
5	10	
6	13	
Total	145	

Miscellaneous reference notes

Selected C++ operator functions:

```
class_name & operator= ( class_name other);
class_name & operator= ( const class_name& other);
std::ostream & operator<<(std::ostream& output, const class_name& obj);
std::istream & operator>>(std::istream& input, class_name& obj);
class_name operator+(const class_name& b);
class_name operator-(const class_name& b);
bool operator==(const class_name& lhs, const class_name& rhs);
```

NAME: _____

1.) [12] Answer the following questions about the **C language** briefly (a sentence or two at most).

(a) [2 points] Is it possible to pass arguments by reference in C?

(b) [2 points] How do you declare a statically allocated two-dimensional array in **C** that can hold a 10-row by 5-column matrix of char values?

(c) [4 points] Will the following file containing **C** code compile and link successfully with the "`gcc main.c`" command to produce an executable? If not, why not?

```
/* file: foo.c */
a = 3.1;
void print() { printf("%d",a); }
```

(d) [4 points] In **C**, what does the & operator do when used in front of a variable, e.g., in an expression such as **"`y = &x;`"**? Give a simple code fragment example to illustrate your answer (make sure to declare all variables in your example).

NAME: _____

2.) [**30 points**] Write a complete **C program** that prompts the user for a single sentence (terminated with one of .!?) as input and wraps lines to 20 characters without breaking up words. This means that each line should be at most 20 characters (not counting the newline character). You can assume that words are separated by single spaces. For example:

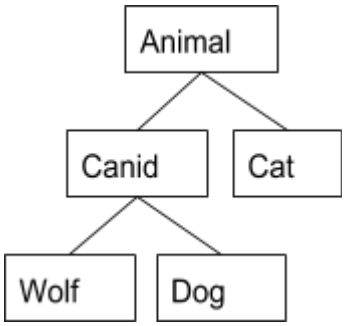
```
$ ./myprog
Please enter a sentence: Did you know that the first digital computer
game never made any money?
Output:
Did you know that
the first digital
computer game never
made any money?
```

You can assume that the user input will not exceed 200 characters. Do not worry about including standard headers if you don't remember them exactly (no points will be deducted for missing includes).

NAME: _____

NAME: _____

3.) [**45 points**] The language for these questions is **C++.** Define a class hierarchy as illustrated below and with the following specifications:



- The Animal class should be abstract, i.e., no non-pointer variables of type Animal can ever be declared.
- Each class in the hierarchy should contain the **`makeSound()`** method, which has no return value or arguments.
- The Cat and Dog classes should have a string name data member.
- The implementation of the **`makeSound()`** method in the Dog, Wolf, and Cat classes should output an appropriate message to standard output, such as “Wendell says: Woof!” (for a Dog named Wendell), “Howl” (for Wolf), and “Xena says: Meow!” (for a Cat named Xena).

a) **[15pts]** Write the implementations of all these classes below. All the code can be in each class’s definition in a single file **animals.hpp** (no need for a separate animals.cpp file).

NAME: _____

3.b) [15pts] Write a main program that:

- **i)** Creates 5 Animal (raw or smart) pointers in an appropriate container (e.g., vector or set), then initializes them with: two Cat, two Dog, and one Wolf objects.
- **ii)** Calls the `makeSound()` method on each container element. Do not use a loop (such as **for** or **while**). If you do not know how to do this without a loop, just write the loop for partial credit.

3.c) [15pts] Overload the **<<** operator to print the same information as the **makeSound()** method for each of the classes that implement makeSound() in part a). Then, write a new version of main() that uses the overloaded **<<** operator instead of calling makeSound().

NAME: _____

4.) [35 points] In this problem you will use both **C and C++**. Do not worry about including standard headers if you don't remember them exactly (no points deducted for missing includes).

4.1) [20 points] Implement a String type in C using a struct that contains a ***dynamically allocated*** **`char`** array. The following functions should be implemented (you may also decide to add more functions of your choosing); unless otherwise specified, you must decide what the arguments are and what each function should return.

- create: creates a String of a specified size passed as an **int** input parameter
- copy: copies a string specified as one of the arguments (e.g., **`const String`** or a pointer, **`String *s`** -- you choose)
- clear: frees all memory for the String

(a)[15 pts] Implement the C String struct and the above functions that operate on it.

(b)[5 pts] Write a main program that uses all three (create, copy, clear) functions correctly.

(More space for problem 4.1).

NAME: _____

4.2) [15 points] Now implement a String **class** in **C++**, which also uses a ***dynamically allocated*** char array to store the characters (**do not** use std::string or any other std container). Implement the same functionality as you did in 4.1 in the following functions:

- constructor: creates a String of a specified size passed as an **int** input parameter
- overloaded assignment operator: copies a string specified as one of the arguments (**const String& s**)
- destructor: frees all memory for the String

(a)[10 pts] Implement the C++ String class as described above.

(b)[5 pts] Write a main program that uses all the functionality you implemented in the String class.

NAME: _____

5.) [10 points] These questions are about the **C++ language.** Consider the following code:

```
#include <iostream>
#include <utility>    // std::pair

class MyInfo {
public:
    friend std::istream& operator>>(std::istream& is, MyInfo &obj) {
      is >> obj.p.second >> obj.p.first;
      return is;
    }

    friend std::ostream& operator<<(std::ostream& os, const MyInfo &obj) {
      os << "My info: " << obj.p.first << " = " << obj.p.second;
      return os;
    }

    std::pair<std::string,int> p;
};

int main() {
    MyInfo m;
    std::cin >> m;
    std::cout << m;
}
```

(a) [5 points] You compile and run this program, then enter “15 Apples” as your input. **What output is produced?**

(b) [2 points] Why are the << and >> overloaded operators defined using the keyword **friend**?

(c) [3 points] What are **three C++ operators** that cannot be overloaded? (there are more than three, but you only need to mention three of them):

NAME: _____

6.) [13] Specify the **Unix shell (bash) command(s)** for each task below (multiple correct answers are possible, just give one for each part).

(a) [2 points] List *all* files in the current directory, displaying the files' permissions.

(b) [2 points] List/view processes running on the machine.

(c) [2 points] Kill a process with PID 425.

(d) [2 points] Delete all files in the current directory.

(e) [2 points] Check if the **`libpng.so`** library is available on ix-dev and find its path.

(f) [2 points] Rename the file **`myprog.c`** `to` **`main.c`** in the current directory.

(g) [1 point] Find the full path of the current directory.

NAME: _____

NAME: _____