[**Open book, handouts, and notes**] Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. You should answer all 3 questions. Due to time limitations, you are not required to comment your code on this exam. Note that the exam is double sided.

1. Consider the following Time class definition:

```
   #include <iostream>
2  using std::ostream;
   using std::istream;
4
   class Time {
6  public:
     Time();
8    Time(unsigned int hr, unsigned int min=0, double sec=0.0);
     ~Time();
10   Time operator+(const Time& rhs);
     Time operator-(const Time& rhs);
12   Time operator-(int min);
     bool read(istream& is);
14   void display(ostream& os);
   private:
16   unsigned int hour;
     unsigned int minute;
18   double second;
   };
20 ostream& operator<<(ostream& os, const Time& time);
   istream& operator>>(istream& is, Time& time);
```

(a) (10 points) List the line numbers for all of the functions that should have been defined as `const` functions. (For partial credit, be sure to explain your reasoning.)

(b) (15 points) Write a program that makes use of the Time class that will:

- Create a `const` Time object that indicates a bedtime of 10:00pm.
- Asks the user to enter an appointment time.
- Indicates the time of day one hour after the appointment begins.
- Displays the amount of time between the appointment time and bedtime (assume that the appointment occurs before bedtime).

(c) (5 points) For each data member from the Time class, determine a reasonable range of values for which the data member may contain.

(d) (10 points) Implement the constructor used in part **b** to declare a Time object representing bedtime. You may assume that all of the other member functions have been implemented.

(e) (10 points) Implement the extraction operator (line 21). You may assume that all of the other member functions have been implemented.

(f) (10 points) Implement the - operator for integers (line 12). You may assume that all of the other member functions have been implemented.

1. (20 points) Write the header file associated with a Complex class that will allow Complex objects to be declared and used as shown in the code below. Note: You should indicate all the appropriate preprocessor directives (#include files, etc). Write **only** the function **prototypes** for the functions **used in the code below**. You **do not** need to **implement** the functions.

```
#include "Complex.h"

int main()
{
  Complex a;
  Complex b(0,1);   // b = 0 + i
  Complex c=2.0;    // c = 2.0 + i0
  std::cout << "Enter a complex number ";
  std::cin >> a;
  c = 3.2 + b;
  std::cout << c;
  c += a;
  c = b - 7.25;
  a = c + 0.52;

  return 0;
}
```

3. (20 points) If any errors are present in the following code, identify the line where the first error occurs, and draw a diagram (like the ones done in lecture) to describe the memory conditions just prior to the error. If no errors are present, draw a diagram to describe the memory conditions after lines 1-15 have been executed.

```
int main()
{
  char y[] = "Give a skeptic a pinch and...";
  int i = 16;
  int* ptr2;
  char* ptr3 = 0;
  int* ptr4 = &i;
  char* ptr5 = &y[*ptr4];
  *ptr5 = y[19];
  ++ptr5;
  char x[]="He will measure it.";
  ptr3 = x;
  ptr3 += 2;
  *ptr5 = *ptr3;

  return 0;
}
```

Additional work area for any problem. Clearly identify to which problem the work on this page is related.