1. *Write a program that converts from 24-hour notation to 12-hour notation. For example, it should convert 14:25 to 2:25 PM. The input is given as two integers. There should be at least three functions, one for input, one to do the conversion, and one for output. Record the AM/PM information as a value of type char, 'A' for AM and 'P' for PM. Thus, the function for doing the conversions will have a call-by-reference formal parameter of type char to record whether it is AM or PM. (The function will have other parameters as well.) Include a loop that lets the user repeat this computation for new input values again and again until the user says he or she wants to end the program.*

2. *Write a program that requests the current time and a waiting time as two integers for the number of hours and the number of minutes to wait. The program then outputs what the time will be after the waiting period. Use 24-hour notation for the times. Include a loop that lets the user repeat this calculation for additional input values until the user says she or he wants to end the program.*

3. *Write a program that inputs a date (for example, July 4, 2008) and outputs the day of the week that corresponds to that date. The following algorithm is from* http://en.wikipedia.org/wiki/Calculating_the_day_of_the_week.

   The implementation will require several functions.

   ```
   bool isLeapYear(int year);
   ```

   This function should return true if year is a leap year and false if it is not. Here is pseudocode to determine a leap year: leap_year = (year divisible by 400) or (year divisible by 4 and year not divisible by 100))

   ```
   int getCenturyValue(int year);
   ```

   *This function should take the first two digits of the year (that is, the century),divide by 4, and save the remainder. Subtract the remainder from 3 and return this value multiplied by 2. For example, the year 2008 becomes: (20/4) = 5 with a remainder of 0. 3 − 0 = 3. Return 3 * 2 = 6.*

   ```
   int getYearValue(int year);
   ```

   This function computes a value based on the years since the beginning of the century. First, extract the last two digits of the year. For example, 08 is extracted for 2008. Next, factor in leap years. Divide the value from the previous step by 4 and discard the remainder. Add the two results together and return this value. For example, from 2008 we extract 08. Then (8/4) = 2 with a remainder of 0. Return 2 + 8 = 10.

   ```
   int getMonthValue(int month, int year);
   ```

   This function should return a value based on the table below and will require invoking the isLeapYear function.

Month	Return Value
January	0 (6 if year is a leap year)
February	3 (2 if year is a leap year)
March	3
April	6
May	1
June	4
July	6
August	2
September	5
October	0
November	3
December	5

   Finally, to compute the day of the week, compute the sum of the date's day plus the values returned by getMonthValue, getYearValue, and getCenturyValue. Divide the sum by 7 and compute the remainder. A remainder of 0 corresponds to Sunday, 1 corresponds to Monday, etc., up to 6, which corresponds to Saturday. For example, the date July 4, 2008 should be computed as (day of month) + (getMonthValue) + (getYearValue) + (getCenturyValue) = 4 + 6 + 10 + 6 = 26. 26/7 = 3 with a remainder of 5. The fifth day of the week corresponds to Friday. Your program should allow the user to enter any date and output the corresponding day of the week in English.

This program should include a `void` *function named* `getInput` *that prompts the user for the date and returns the month, day, and year using pass-by-reference parameters. You may choose to have the user enter the date's month as either a number (1–12) or a month name.*

4. *Hexadecimal numerals are integers written in base 16. The 16 digits used are '0' through '9' plus 'a' for the "digit 10", 'b' for the "digit 11", 'c' for the "digit 12", 'd' for the "digit 13", 'e' for the "digit 14", and 'f ' for the "digit 15". For example, the hexadecimal numeral d is the same as base 10 numeral 13 and the hexadecimal numeral 1d is the same as the base 10 numeral 29. Write a C++ program to perform addition of two hexadecimal numerals each with up to 10 digits. If the result of the addition is more than 10 digits long, then simply give the output message "Addition Overflow" and not the result of the addition. Use arrays to store hexadecimal numerals as arrays of characters. Include a loop to repeat this calculation for new numbers until the user says she or he wants to end the program.*

5. *Write a program that reads in a list of integers into an array with base type* `int`*. Provide the facility to either read this array from the keyboard or from a file, at the user's option. If the user chooses file input, the program should request a file name. You may assume that there are fewer than 50 entries in the array. Your program determines how many entries there are. The output is to be a two-column list. The first column is a list of the distinct array elements; the second column is the count of the number of occurrences of each element. The list should be sorted on entries in the first column, largest to smallest*

   For example, for the input
   -12 3 -12 4 1 1 -12 1 -1 1 2 3 4 2 3 -12
   the output should be

N	*Count*
4	*2*
3	*3*
2	*2*
1	*4*
-1	*1*
-12	*4*