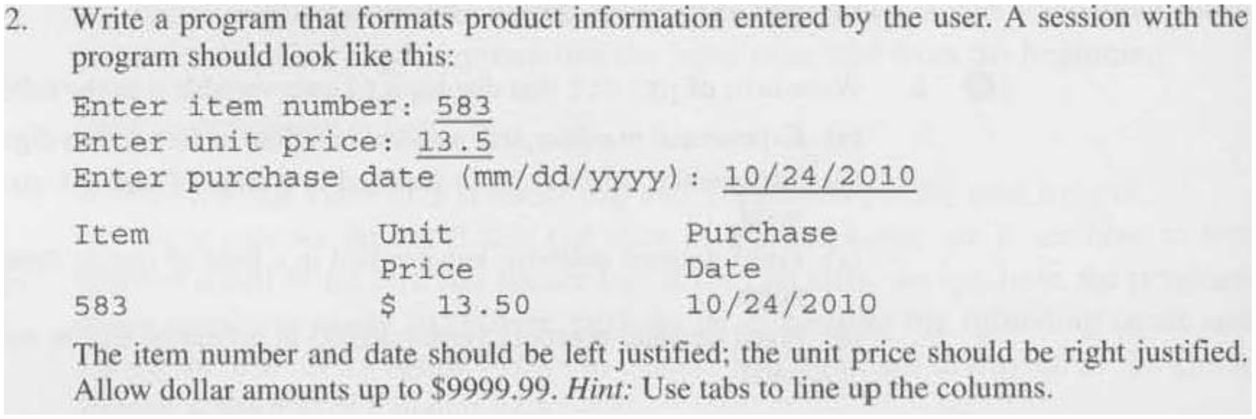
York University
Electrical Engineering and Computer Science

EECS2031: Software Tools
SU2016
Assignment #2

Chapter 3: Programming Projects

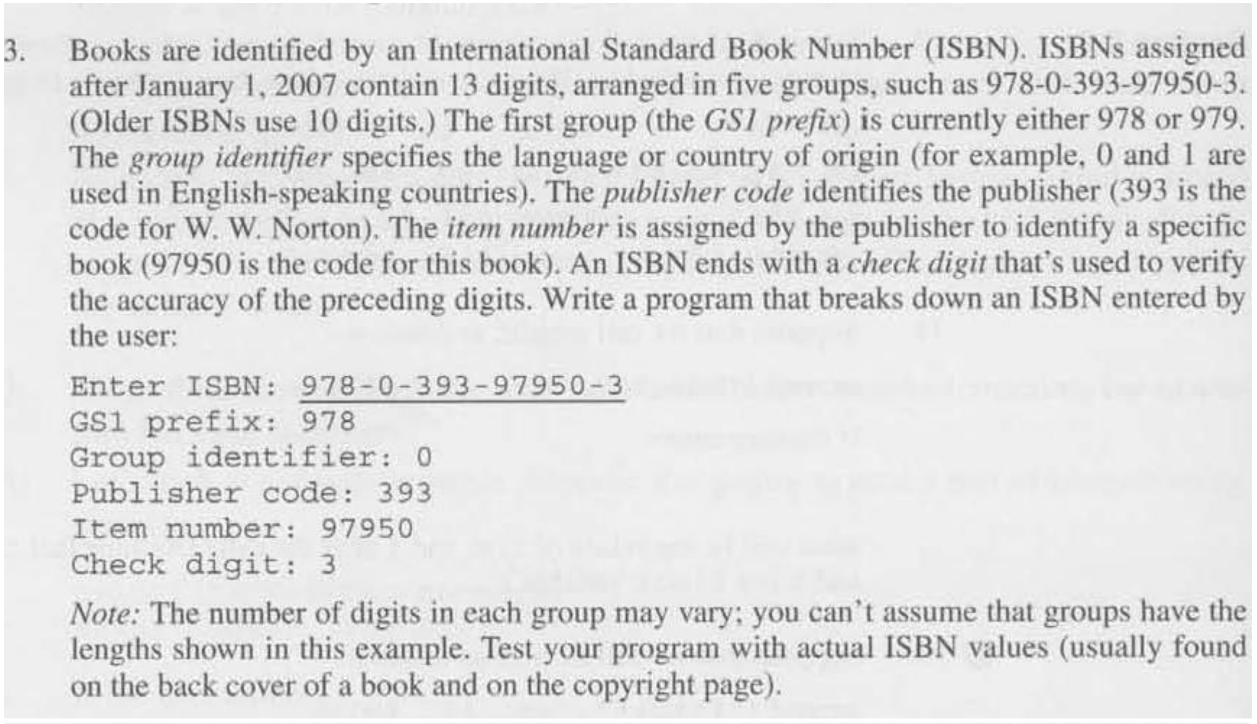


2. Write a program that formats product information entered by the user. A session with the program should look like this:

```
Enter item number: 583
Enter unit price: 13.5
Enter purchase date (mm/dd/yyyy): 10/24/2010

Item            Unit            Purchase
                Price           Date
583             $   13.50       10/24/2010
```

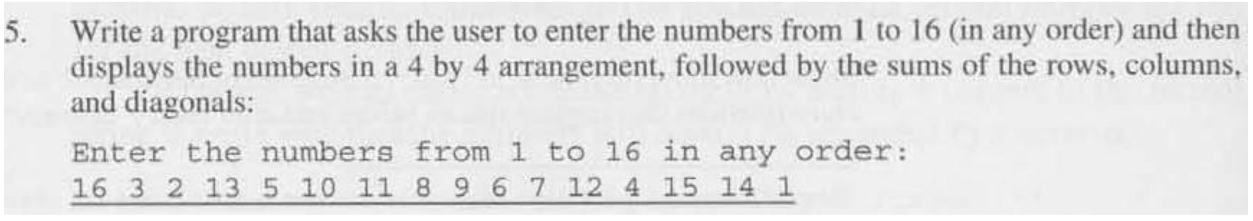
The item number and date should be left justified; the unit price should be right justified. Allow dollar amounts up to $9999.99. *Hint:* Use tabs to line up the columns.



3. Books are identified by an International Standard Book Number (ISBN). ISBNs assigned after January 1, 2007 contain 13 digits, arranged in five groups, such as 978-0-393-97950-3. (Older ISBNs use 10 digits.) The first group (the *GS1 prefix*) is currently either 978 or 979. The *group identifier* specifies the language or country of origin (for example, 0 and 1 are used in English-speaking countries). The *publisher code* identifies the publisher (393 is the code for W. W. Norton). The *item number* is assigned by the publisher to identify a specific book (97950 is the code for this book). An ISBN ends with a *check digit* that's used to verify the accuracy of the preceding digits. Write a program that breaks down an ISBN entered by the user:

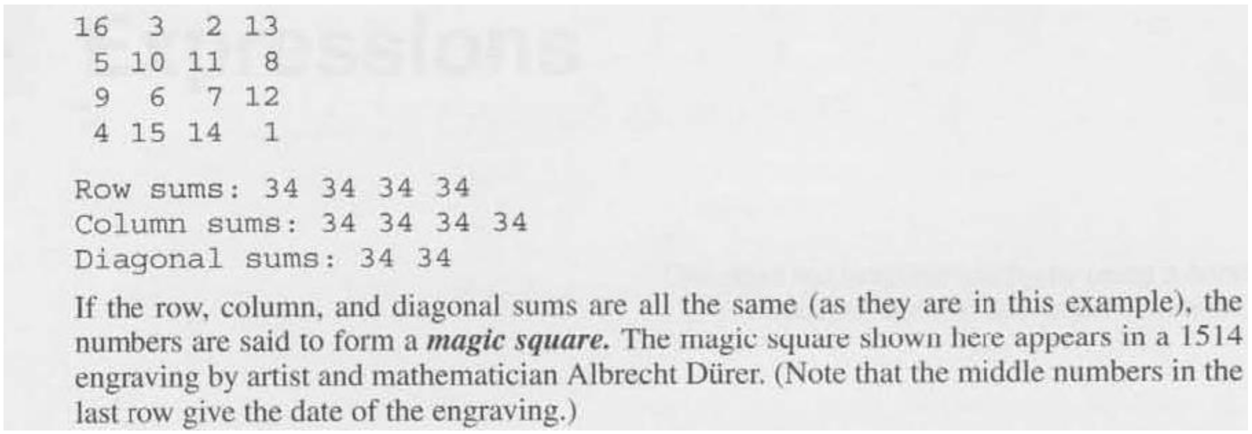
```
Enter ISBN: 978-0-393-97950-3
GS1 prefix: 978
Group identifier: 0
Publisher code: 393
Item number: 97950
Check digit: 3
```

Note: The number of digits in each group may vary; you can't assume that groups have the lengths shown in this example. Test your program with actual ISBN values (usually found on the back cover of a book and on the copyright page).



5. Write a program that asks the user to enter the numbers from 1 to 16 (in any order) and then displays the numbers in a 4 by 4 arrangement, followed by the sums of the rows, columns, and diagonals:

```
Enter the numbers from 1 to 16 in any order:
16 3 2 13 5 10 11 8 9 6 7 12 4 15 14 1
```

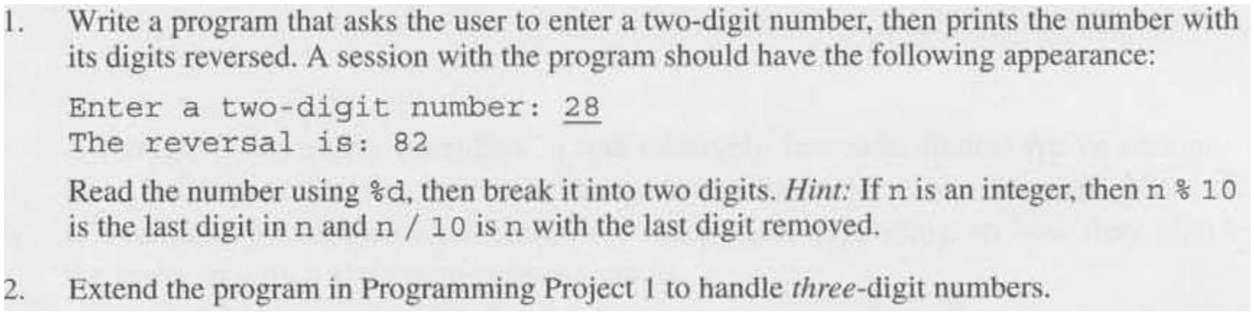


```
16  3  2 13
 5 10 11  8
 9  6  7 12
 4 15 14  1

Row sums: 34 34 34 34
Column sums: 34 34 34 34
Diagonal sums: 34 34
```

If the row, column, and diagonal sums are all the same (as they are in this example), the numbers are said to form a ***magic square.*** The magic square shown here appears in a 1514 engraving by artist and mathematician Albrecht Dürer. (Note that the middle numbers in the last row give the date of the engraving.)

Chapter 4: Programming Projects



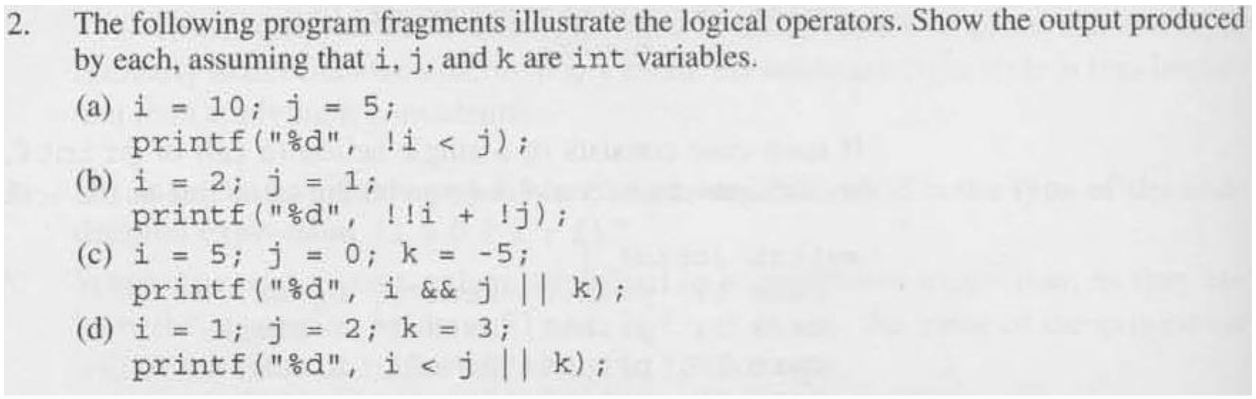
1. Write a program that asks the user to enter a two-digit number, then prints the number with its digits reversed. A session with the program should have the following appearance:

```
Enter a two-digit number: 28
The reversal is: 82
```

Read the number using `%d`, then break it into two digits. *Hint:* If `n` is an integer, then `n % 10` is the last digit in `n` and `n / 10` is `n` with the last digit removed.

2. Extend the program in Programming Project 1 to handle *three*-digit numbers.

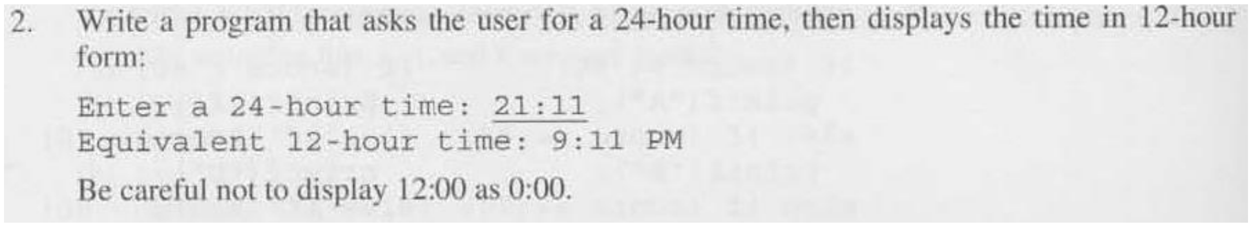
Chapter 5: Exercises



2. The following program fragments illustrate the logical operators. Show the output produced by each, assuming that `i`, `j`, and `k` are `int` variables.

```
(a) i = 10; j = 5;
    printf("%d", !i < j);
(b) i = 2; j = 1;
    printf("%d", !!i + !j);
(c) i = 5; j = 0; k = -5;
    printf("%d", i && j || k);
(d) i = 1; j = 2; k = 3;
    printf("%d", i < j || k);
```

Chapter 5: Programming Projects



2. Write a program that asks the user for a 24-hour time, then displays the time in 12-hour form:

```
Enter a 24-hour time: 21:11
Equivalent 12-hour time: 9:11 PM
```

Be careful not to display 12:00 as 0:00.

4. Here's a simplified version of the Beaufort scale, which is used to estimate wind force:

Speed (knots)	*Description*
Less than 1	Calm
1–3	Light air
4–27	Breeze
28–47	Gale
48–63	Storm
Above 63	Hurricane

Write a program that asks the user to enter a wind speed (in knots), then displays the corresponding description.

8. The following table shows the daily flights from one city to another:

Departure time	*Arrival time*
8:00 a.m.	10:16 a.m.
9:43 a.m.	11:52 a.m.
11:19 a.m.	1:31 p.m.
12:47 p.m.	3:00 p.m.
2:00 p.m.	4:08 p.m.
3:45 p.m.	5:55 p.m.
7:00 p.m.	9:20 p.m.
9:45 p.m.	11:58 p.m.

Write a program that asks user to enter a time (expressed in hours and minutes, using the 24-hour clock). The program then displays the departure and arrival times for the flight whose departure time is closest to that entered by the user:

```
Enter a 24-hour time: 13:15
Closest departure time is 12:47 p.m., arriving at 3:00 p.m.
```

Hint: Convert the input into a time expressed in minutes since midnight, and compare it to the departure times, also expressed in minutes since midnight. For example, 13:15 is $13 \times 60 + 15 = 795$ minutes since midnight, which is closer to 12:47 p.m. (767 minutes since midnight) than to any of the other departure times.