ENCMP 100 – Computer Programming for Engineers
Assignment #2
Due: Monday, Feb. 13 , 2017 at 6:00pm MST

Objective

This assignment is designed to provide you with practice using selection statements to solve an encoding problem. There are many applications for encoding and decoding information in engineering contexts. For example, in the area of Computer Engineering there is often a need to transmit information from one place to another, e.g. the Internet.

In some situations, the information must be converted into a form suitable for transmission from the source to the destination. The encoded information must then be decoded at the destination before it can be used. There are several ways to encode and decode information, e.g., some are designed for secure data transmission, while others decrease the size of the data to be stored or transmitted (compression).

Marking Scheme

You will get a total of 50 points for completing the following:

TASK	POINTS
Successful testing of Rules	35
Quality of code	15
TOTAL	50

Points for Quality of Code
- Complete file header (see Hints for an example) – 3 point
- Design (appropriate use and naming of variables) – 5 points
- Comments in the code – 5 points
- Layout (indentation/spacing) – 2 point

Submission

- Filename to be used for this assignment Assign2_<UofA_ID_Number>.m
        Ex. U of A ID Number: 1234567 filename for assignment #2 is
        Assign2_1234567.m
- Submit only your **.m** file under Assignment 2 in your eClass/Moodle account.
- The assignment is due on Monday, Feb. 13 2017 at 6:00 pm MST

Background

Your friend Maverick recently landed a lucrative engineering co-op work term in an oil-rich country. Meanwhile, you chose to make a modest salary as a co-op intern with the [Canadian Security Intelligence Service](#) (CSIS). A week after arriving in the foreign country, Maverick is kidnapped by rebels opposed to oil exploration. He is held captive in a small village within rebel territory. Luckily, a group of disenchanted rebels are willing to smuggle Maverick to one of seven rendezvous points:

- the village bridge
- the village library
- river crossing
- nearby airport
- bus terminal
- hospital
- at St. Pete's Church

where an allied helicopter will pick them up and fly them to safety. For the plan to work, the friendly rebels need to know two pieces of information:

- the day of the rendezvous
- the rendezvous point

CSIS has devised a nine-digit code to carry this information to the friendly rebels. When the time is ripe, CSIS will insert a coded message into the online newspaper of the oil-rich country, as a number on the front page. The rebels have Internet access via satellite links and follow the news. While it is helpful (as a decoy strategy) that the newspaper has other numbers on the front page, e.g. in real adverts, it is vital that the rebel insiders can detect and decode the secret message. CSIS has found a way to smuggle a decoder to the friendly rebels. You are asked to program the decoder in MATLAB. The safety of your friend Maverick depends on your ability to program the decoder correctly.

Details

The four rules for detecting and decoding the code are as follows:

1. A valid code must be a 9-digit number.
     e.g.   123456789, message MAY be valid
     e.g.   12345678, message invalid
     e.g.   1234567890, message invalid

   Assume that the input data contains only digits, with a non-zero first digit.

2. The code must pass the odd-even "truth" test.

   - If the sum of the digits is odd, the message is invalid.
      e.g. 222222223, sum = 19, message invalid
  - If the sum of the digits is even, the message MAY be valid.
      e.g. 222222222,, sum = 18, message may be valid

3. To determine the rescue day, multiply the 1st digit by the 3rd digit, then subtract the 5th digit. The answer indicates the rescue day, as follows:

     1 = Monday   2 = Tuesday   3 = Wednesday
     4 = Thursday 5 = Friday   6 = Saturday   7 = Sunday

 Any other number indicates that the message is invalid.

     Examples: 234567890   `2x4-6 = 2`   Rescue is on Tuesday
             712245420   `7x2-4 = 10`   Message is invalid.

4. The last three digits determine the rendezvous point.

- If the following result when you multiply the 2nd digit by the 4th digit, then subtract the 6th digit is divisible by 3, then the rendezvous point is
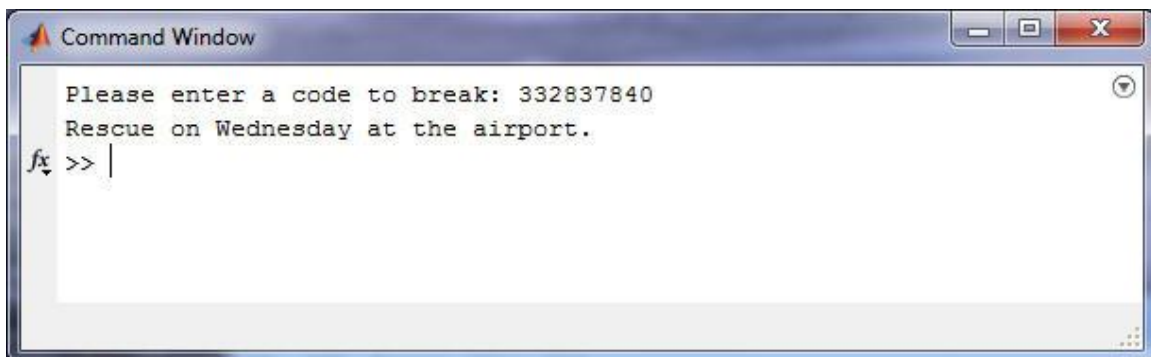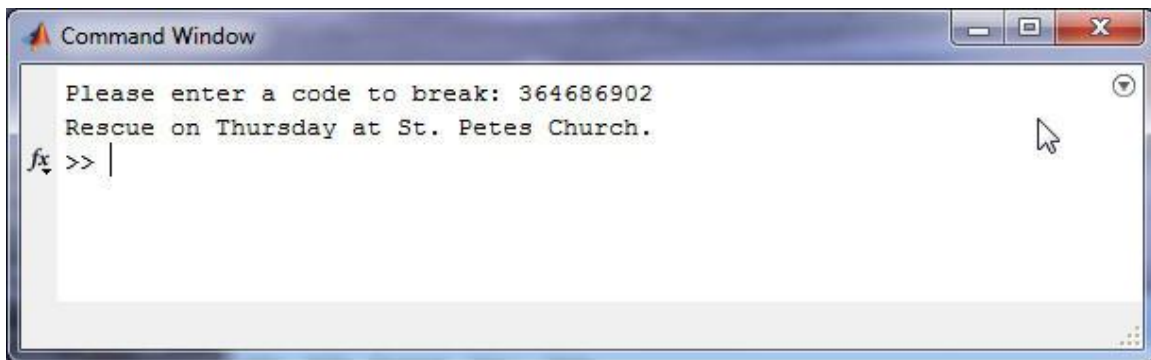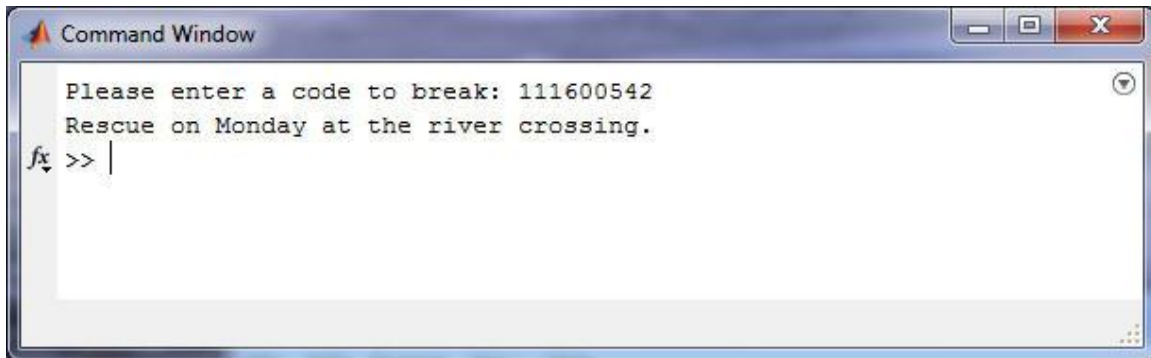   7th digit - 9th digit
- If the following result when you multiply the 2nd digit by the 4th digit, then subtract the 6th digit is not divisible by 3, then the rendezvous point is
          8th digit - 9th digit

Rendezvous Number	Rendezvous Point
1	bridge
2	library
3	river crossing
4	airport
5	bus terminal
6	hospital
7	St. Petes Church

If the result is one of the following rendezvous numbers, the message is valid:

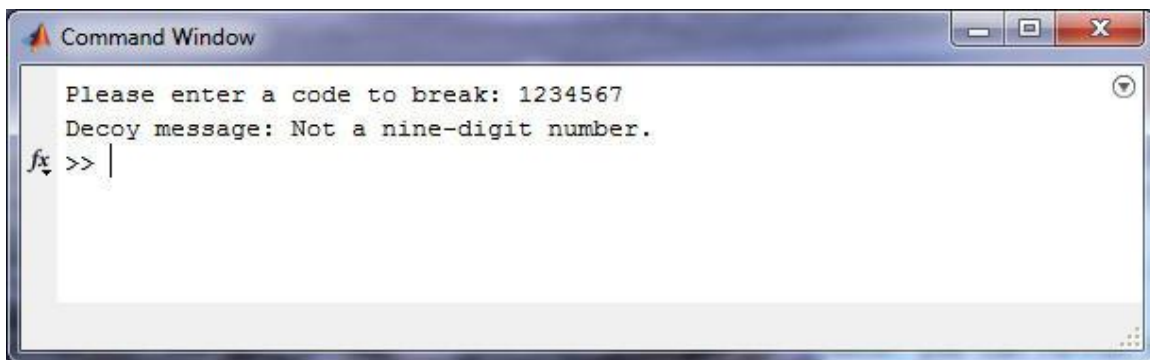Any other rendezvous number means that the message is invalid

Output

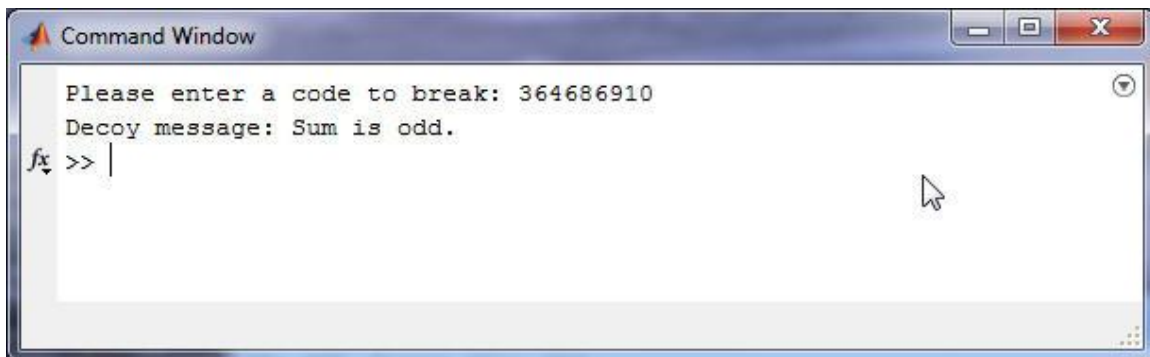Your output should look like the example screenshots below. Here are 3 valid messages:

```
Command Window

Please enter a code to break: 111600542
Rescue on Monday at the river crossing.
fx >> |
```

```
Command Window

Please enter a code to break: 364686902
Rescue on Thursday at St. Petes Church.
fx >> |
```

```
Command Window

Please enter a code to break: 332837840
Rescue on Wednesday at the airport.
fx >> |
```

Your output for invalid messages should indicate only the <u>first</u> reason encountered for rejecting the message. The reasons are one of the following, in this order:

- Decoy Message: Not a nine-digit number.
- Decoy Message: Sum is odd.
- Decoy Message: Invalid rescue day.
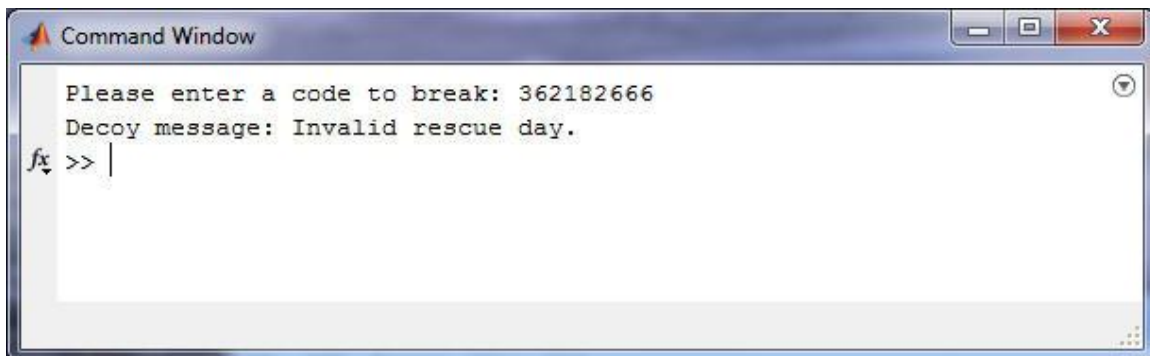- Decoy Message: Invalid rendezvous point.

Command Window

```
   Please enter a code to break: 1234567
   Decoy message: Not a nine-digit number.
fx >> |
```
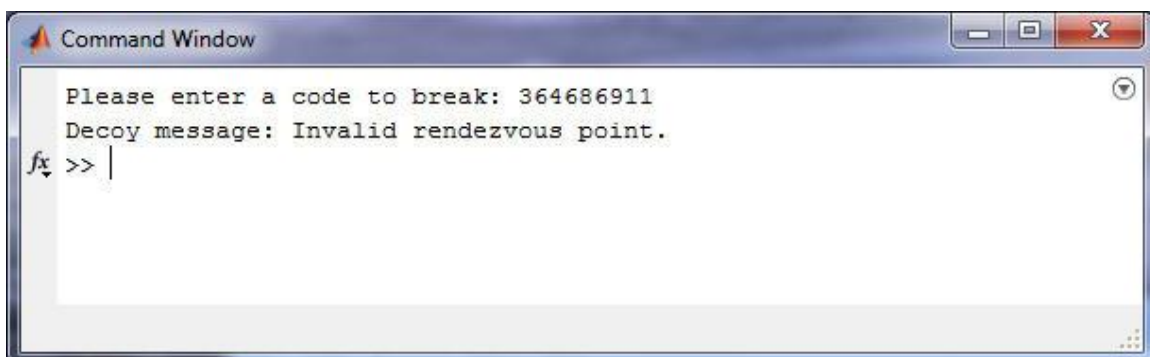
Command Window

```
   Please enter a code to break: 364686910
   Decoy message: Sum is odd.
fx >> |
```

Command Window

```
   Please enter a code to break: 362182666
   Decoy message: Invalid rescue day.
fx >> |
```

Command Window

```
   Please enter a code to break: 364686911
   Decoy message: Invalid rendezvous point.
fx >> |
```

Code Requirements

You are required to use **two switch statements** in this assignment, one to determine the rescue day and another to determine the rendezvous point. Failure to use two switch statements in this assignment will result in a mark of zero on Successful testing of the rules portion of the mark breakdown.

Hints

1. Before you even begin to write out any MATLAB code, sit down and sketch out your program design in abbreviated English. This kind of high-level program design is called pseudo code. Please see secondary link for full pseudo code of the assignment two. It would start off something like this:

Get secret code from user as a string
Get an array of the individual digits
If *code not 9 digits*
   Output error message
Else
   Get sum of digits
   If *sum is odd*
      Output error message
   Else
      Get rescue day
      Etc, ...

Notice how the logic naturally translates into an `if-else` chain. The `if` clauses in the chain contain messages for invalid codes. After such a message is output, the control flow breaks out of the entire chain. You end up reaching the final `else` clause in the chain only if the code is valid. It is there that the valid message is composed and output.

For the complete pseudo code for this assignment, please see second link on eClass site under Assignment #2. Please note that the posted pseudo code is a "first pass" attempt of what you should do when creating pseudo code for your assignments.

2. To help you read in a secret code from the user as a string, type in help input on the command window or search the input command using the help function.

3. When converting an array of characters (string) to an array of numbers, you need to subtract the ASCII value of zero from each character. For example, if your string is called `code_str` and your number array is called `digits`, you can do the following:

```
digits = code_str - '0';
```

4. Remember to use the built-in MATLAB functions. For example, to check if the string contains six digits, you might find the **length** function helpful. To find the sum of the digits, you might find the **sum** function helpful. To check if the sum of digits is odd, you may find the **mod** function helpful.

5. Build up your program design in stages. After you have completed the code for Rule 1, compile and test with a variety of codes that satisfy and fail to satisfy the rule. Then go on to write the code for the next rule, etc.

6. Please remember to include your header at the top of your .m file.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Course: ENCMP 100
% Assignment: 2
% Name: Joe MacDonald
% CCID: jmac
% U of A ID: 1234567

% Acknowledgements:
%I received help from Jason Smith on using the mod function.
%
% Description:
% This program contains four rules which can be used to crack the
% secret code to save Maverick the co-op student.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```