Hello,

Thanks for taking the time out to attempt the assignment. This assignment is designed to test algo and coding skills. We expect you to write pseudo-code with proper comments in a txt file and send it across to us. Do mention the worst case time and space complexity of your solution. We'll reach out to you shortly post that. Thanks.

Task 1 URLs are strings with a simple syntax:

scheme://[username:password@]domain[:port]/path?query_string#fragment_id

Parse a well-formed URL to retrieve the relevant information: scheme, domain, path, ...

According to the standards, the characters:

! * ' ( ) ; : @ & = + $, / ? % # [ ]
only need to be percent-encoded (%) in case of possible confusion.

Also note that the path, query and fragment are case sensitive, even if the scheme and domain are not.

The way the returned information is provided (set of variables, array, structured, record, object,...) is language-dependent and left to the programmer, but the code should be clear enough to reuse.

Extra credit is given for clear error diagnostics.

Here is the official standard: https://tools.ietf.org/html/rfc3986,
and here is a simpler BNF: http://www.w3.org/Addressing/URL/5_URI_BNF.html.

Test cases

foo://example.com:8042/over/there?name=ferret#nose should parse into:

scheme = foo
domain = example.com
port = :8042
path = over/there
query = name=ferret

fragment = nose

urn:example:animal:ferret:nose   should parse into:

scheme = urn
path = example:animal:ferret:nose

other URLs that must be parsed include:

jdbc:mysql://test_user:ouupppssss@localhost:3306/sakila?profileSQL=true
ftp://ftp.is.co.za/rfc/rfc1808.txt
http://www.ietf.org/rfc/rfc2396.txt#header1
ldap://[2001:db8::7]/c=GB?objectClass=one&objectClass=two
mailto:John.Doe@example.com
news:comp.infosystems.www.servers.unix
tel:+1-816-555-1212
telnet://192.0.2.16:80/
urn:oasis:names:specification:docbook:dtd:xml:4.1.2

Task 2 Flip the bits

Given an N by N square array of zeros or ones in an initial configuration, and a target configuration of zeroes and ones The task is to transform one to the other in as few moves as possible by inverting whole numbered rows or whole lettered columns at once, as one move.

In an inversion any 1 becomes 0, and any 0 becomes 1 for that whole row or column.

Task
Create a program to score for the game.

The game should create an original random target configuration and a starting configuration.
Ensure that the starting position is never the target position.
The target position must be guaranteed as reachable from the starting position.
(One possible way to do this is to generate the start position by legal flips from a random target position. The flips will always be reversible back to the target from the given start position).
The number of moves taken so far should be shown.

Show an example of a short game here, on this page, for a 3 by 3 array of bits

```
A> flip 3
Please press some keys to generate a random state...done.

*** FLIP THE BITS ***
---------------------



Your flips: 0   Goal: 20
--Board------------     --Goal-------------
  A B C                A B C
1 0 0 0              1 0 1 1
2 0 0 0              2 1 0 0
3 1 1 1              3 0 1 1

Press line or column to flip, or Q to quit: a

Your flips: 1   Goal: 20
--Board------------     --Goal-------------
  A B C                A B C
1 1 0 0              1 0 1 1
2 1 0 0              2 1 0 0
3 0 1 1              3 0 1 1
Press line or column to flip, or Q to quit: 1

You win!
```

Task 3 Climb Stairs

There's a staircase with N steps, and you can climb 1 or 2 steps at a time. Given N, write a function that returns the number of unique ways you can climb the staircase. The order of the steps matters.

For example, if N is 4, then there are 5 unique ways:

1, 1, 1, 1

2, 1, 1

1, 2, 1

1, 1, 2

2, 2

What if, instead of being able to climb 1 or 2 steps at a time, you could climb any number from a set of positive integers X? For example, if X = {1, 3, 5}, you could climb 1, 3, or 5 steps at a time. Generalize your function to take in X.