Assignment 1: Ethical and Robust Web Crawler

(Due March 9 at 4:30pm)

Write a **Python web crawler** that is both **ethical** and **robust**. Your crawler must start from the following **seed URL**: https://informationsystems.umbc.edu/. For **each eligible web page you download**, please identify: the **final URL** (after redirects, if any) and the **page title** (from the HTML <title> tag). Save them to a CSV file named: *assignment1_res_<lastname>.csv* , which contains a header row and the following columns: url and title. Each row corresponds to **one unique eligible page**.

To avoid runaway crawling, only crawl pages whose domain ends with *informationsystems.umbc.edu* . Pages outside the scope must **not** be included in your output. Only include **web pages** (Content-Type contains text/html). Do not crawl or include non-HTML resources such as PDFs, images, videos, audio, Word/PowerPoint files, and archives (e.g., .zip).

Your output must not contain duplicate pages. Treat the following as duplicates and keep only one entry:

- URLs that differ only by a fragment (e.g., #section)
- URLs that redirect to a canonical URL (store the final URL)
- Normalize trailing slashes consistently

To prevent runaway crawling, your crawler must implement **at least one** stopping condition, such as: *maximum number of pages visited* or *maximum runtime*.

What to submit

Please submit the following **four** items:

1. **Code**: Please put all the code in a single Python file called *crawler_<lastname>.py*
2. **README**: *README_<lastname>* which includes
    - Python version and required libraries
    - installation commands (e.g., pip install ...)
    - exact run command (e.g., python crawler_<lastname>.py)
    - description of scope rules, stopping condition(s), and ethical considerations
3. **Results**: assignment1_res_<lastname>.csv
4. Answers to the following **two** questions (in a word of pdf file)

    (a) What search strategy does your crawler use to traverse pages: **BFS**, **DFS**, or **Best-First Search**? Explain your answer using evidence from your implementation/output.

(b) How does your crawler handle **dynamic pages**? Please clearly explain your approach.

Grading criteria

Your submission will be assessed based on:

1. **Correctness & reliability of your Python code**
    - code runs without exceptions, crashes or hanging
    - handles errors gracefully
2. **Precision (no incorrect pages)**
    - only eligible HTML pages within the allowed domain are included in the results
    - no disallowed/out-of-scope pages
3. **Recall (coverage)**
    - you did not miss pages that satisfy the criteria (within your scope and stopping limits)
4. **Output quality**
    - clean CSV format with correct URLs and page titles
    - no duplicate entries after normalization
5. **Ethical**
    - robots.txt compliance
    - appropriate request rate and identification