

# CS2315 Computer Programming

## Assignment Three (2021-22 Sem. A)

Deadline: **27-Nov-2021 23:59** (Week 13 Sat)    No Late submission will be accepted

In this assignment you may use *only* the functions from `<iostream>`, `<iomanip>` and `<cstring>` when appropriate (*i.e. no other libraries like <string>, <cmath> or <stdlib>*).

### Parsing of URL

URL (Universal Resource Locator), commonly known as “web address” is a string which specifies the location of a file/resource, together with the (optional) query string. The basic form of a URL is shown as follow:

<code>&lt;protocol&gt; :// &lt;domain&gt; [/&lt;path&gt;] [/&lt;file&gt;] [? &lt;query string&gt;]</code>
---

`<protocol>` indicates the operation. For webpage access, usually it is **http://** (*Hyper Text Transfer Protocol*) or the secure version **https://**

`<domain>` is basically the “server name” like **www.google.com** . It could also be the IP address, possible with port number like **127.0.0.1:80** .

`<path>` is optional, which indicates the directory/folder location of the `<file>` . For instance, in **https://abc.com/def/ghi/file.html** , the `<path>` is **def/ghi** and the `<file>` name is **file.html**

`<file>` name is also optional. For instance **https://abc.com/def/ghi/** contains no `<file>` name (*after the last /* ). Usually the web server will load some default name (*like index.html*) but we will not consider default names in this assignment.

From time to time, the URL may contain (optional) `<query string>` for purpose like login and online form (*e.g. google search*). The query string (which could be empty) is separated from the address using the **?** character.

The `<query string>` contains zero or more parameters in the form of `<name>=<value>` pair. If there exist two or more parameter pairs, they will be separated by the **&** character. For instance, in **https://www.google.com/search?q=cityu&uact=5** After the `<file>` (resource) name **search**, there are 2 parameters: The first one has `<name>` **q** and it's `<value>` is **cityu**, the second one has `<name>` **uact** and it's `<value>` is **5**.

A URL contains no space character, space character (*e.g. in query string*) will be replaced by the **+** character. Also, a character could be replaced by a triplet which contains **%** followed by two hexadecimal digit. For instance, character '**K**' is ASCII code **75** in decimal (**4B** in hexadecimal), therefore '**K**' character could be represented as **K** or **%4b** or **%4B** . Let's try: **https://www.google.com/search?q=H+%4b**

(It will search "**H K**" (<H> <space> <K>))

In this assignment you will finish the implementation of the **URL** class based on a given template and **main()** function.

- Students have to implement all constructors and functions exactly as listed in the table on the next page.
- Students could not change the behavior (*e.g. return type, public/private*) of the stated constructors and functions, but students could add in new data members and functions for internal use if necessary.
- Students could not change the provided **main()** function.
- Students may assume that:
  - ✓ the given URL is syntactically valid. No error checking is needed
  - ✓ the URL length is at most 1024 characters
  - ✓ characters **:** **/** **?** **&** and **=** are not represented in hex encoding (*except in parameter names and values*)
  - ✓ The protocol field is not represented in hex encoding
  - ✓ the URL and query string do not contain foreign character (e.g. Chinese)
  - ✓ parameter names in query string are unique
  - ✓ a parameter may have empty value but the parameter name itself is non-empty
  - ✓ the last word after domain name is treated as **<file>** if it's not preceded by **/**.  
For instance, in **http://abc.com/def/ghi/jkl** , **<path>** is **def/ghi** and **<file>** is **jkl**  
In **http://abc.com/def/ghi/jkl/** , **<path>** is **def/ghi/jkl** and **<file>** is **empty**

# Structure of the URL class:

Students should download the template from Canvas and continue development.

<b>Private data members / methods:</b>	
<b>char str[1025];</b>	cString which stores the content of the URL
<b>Public data members / methods:</b>	
<b>URL()</b>	Default constructor which sets cString <b>str[]</b> to empty
<b>URL(char S[])</b>	Parameterized constructor which copies the input cString <b>S[]</b> to private member <b>str[]</b>
<b>void set(char S[])</b>	Copies the input cString <b>S[]</b> to private member <b>str[]</b>
<b>void printURL()</b>	Display the content of URL. Print "URL is Empty" if <b>str[]</b> has no content ( <i>e.g. just created with default constructor</i> ).
<b>void printURL()</b>	Display the protocol in uppercase Print "URL is Empty" if <b>str[]</b> has no content
<b>void printDomain()</b>	Display the domain/server name Print "URL is Empty" if <b>str[]</b> has no content
<b>void printPath()</b>	Display the path name without leading and trailing / character Print "URL is Empty" if <b>str[]</b> has no content Print "Path is Empty" if URL contains no path
<b>void printFile()</b>	Display the file/resource name Print "URL is Empty" if <b>str[]</b> has no content Print "Filename is Empty" if URL contains no filename
<b>void printAllParam()</b>	Display the number of parameters on the first line, then display each <b>[name]=[value]</b> pair on a line of its own.  Parameters should be sorted by <b>[name]</b> in ascending order  Field width of <b>[name]</b> should be set according to the longest name and the field should be left-aligned.  Print "URL is Empty" if <b>str[]</b> has no content.

## Sample Input / Output:

Example 1: *(User Input is underlined)*

```
Enter URL: https://www.microsoft.com/en-hk/
URL is https://www.microsoft.com/en-hk/
Protocol: HTTPS
Domain: www.microsoft.com
Path: en-hk
Filename is Empty
Number of parameter(s): 0
```

Example 2: *(User Input is underlined)*

```
Enter URL: http://abc.com/def/ghi/index.html?
URL is http://abc.com/def/ghi/index.html?
Protocol: HTTP
Domain: abc.com
Path: def/ghi
Filename: index.html
Number of parameter(s): 0
```

Example 3: *(User Input is underlined)*

```
Enter URL: https://www%2Egoogle.c%6fm/se%61rch?q=cityu+%48k&uact=5
URL is https://www%2Egoogle.c%6fm/se%61rch?q=cityu+%48k&uact=5
Protocol: HTTPS
Domain: www.google.com
Path is Empty
Filename: search
Number of parameter(s): 2
[q ] is [cityu Hk]
[uact] is [5]
```

Example 4: *(User Input is underlined)*

```
Enter URL: https://www.bing.com/search?q=us&qsn&form=QBRE&sp=-1&pq=us&sc=8-2&sk=&cvid=0F
URL is https://www.bing.com/search?q=us&qsn&form=QBRE&sp=-1&pq=us&sc=8-2&sk=&cvid=0F
Protocol: HTTPS
Domain: www.bing.com
Path is Empty
Filename: search
Number of parameter(s): 8
[cvid] is [0F]
[form] is [QBRE]
[pq ] is [us]
[q ] is [us]
[qsn] is [n]
[sc ] is [8-2]
[sk ] is []
[sp ] is [-1]
```

# Marking criteria

Submitted program will be tested repeatedly with PASS. Marks will be graded objectively based on the number of correct outputs reported by PASS.

- If the program is not compilable, zero mark will be given.
- Make sure that the output format (spelling, spacing...etc) follows *exactly* the sample output above otherwise PASS will consider your answer incorrect. Note that the marker will make NO manual attempt to check or re-mark program output.

Unlike tutorial exercises, for assignment, PASS will NOT make ALL test cases visible online. (*i.e. there are hidden test cases*). Email request on opening hidden test case(s) will not be entertained. It is the responsibility of a programmer to design own test cases in order to verify the correctness of the written program.

Note that the marking in PASS is automatic, therefore the following situations may lead to extremely low or zero marks:

- Input/output does not match the requirements as defined.  
(*e.g. incorrect spacing, incorrect spelling, letter cases or punctuations*)
- Submission of non **.cpp** files (*e.g. .exe or .zip files*)
- Submission with the **“test”** function rather than the **“submit”** function  
(*Only the last upload with “submit” will be marked*)

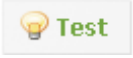
Please also note that the PASS plagiarism check will also be turned on. **The same disciplinary action will be applied without distinguishing which one is the source / copier.** Please safeguard your files if you work on your assignment using public computers (e.g. CityU Lab)

# Testing and Submission

Students should submit **separate** cpp files for **part a)** and **part b)** to PASS before the deadline. **No other submission method (e.g. hardcopy, email...etc) is accepted.** Students may use whatever IDE/compiler for development, but programming using non-standard, platform specific features (which is not compilable in MS Visual Studio) will lead to zero mark. The marker will make no attempt to fix the syntax error and make the code compilable.

**Hint:** If you observe that the answer in PASS is different from the one you get in your PC, most likely it is caused by programming bugs like uninitialized variables or array access with

invalid indexes. The result by PASS will be used for marking without consideration of what you observed in your local PC.

It is advised that students test the programs with PASS (*using the*  *function*) before final submission (*with the “submit” button*). Be warned that the system could be extremely busy and may become sluggish in responding near deadline. Only **.cpp** files are accepted. Do not submit **.obj**, **.exe** or any other files.

To protect yourself, it is advised that you write down your particulars (full name, student number, eid...) in the beginning of your source code *as comment*.