Homework #12 **M1522.000800 System Programming**

Sample Solution

Question 1

HTTP Protocol

Web servers provide content to clients in two different ways: *`static / dynamic contents`*. If web servers run an executable file and return its output to the client, then the output produced by the executable at runtime is a *`dynamic content`*.

Assume that a host whose domain is `www.sysprog.com` provides a dynamic content with executable called `/cgi-bin/something` that will be called with two argument strings:`1234` and `12345`.

a) If we want to conduct transactions to get the output of the dynamic contents using `Unix TELNET`, fill out following HTTP requests. (HTTP Versions are all available.)

```
$ telnet www.sysprog.com 80
Trying xxx.xxx.xxx.xxx...
Connected to www.sysprog.com.
Escape character is '^]'.
GET /cgi-bin/something?1234&12345 HTTP/1.1
Host:www.sysprog.com

```

b) *Chunked transfer encoding* is a data transfer mechanism in `HTTP/1.1` in which data is sent in a series of chunks. It uses the Transfer-Encoding `HTTP` header in place of the Content-Length header, which the earlier version of the `HTTP` would otherwise require. The size of each chunk is sent right before the chunk itself so that the receiver can tell when it has finished receiving data for that chunk. The data transfer is terminated by a final chunk of length zero. Dynamic contents prefer the *chunked transfer encoding* as a data transferring in `HTTP/1.1` and later. Explain why dynamic contents prefer it.

=> *Chunked transfer encoding* allows a server to maintain an `HTTP` persistent connection for dynamically generated content. In this case the `HTTP` *Content-Length header* cannot be used to delimit the content and the next `HTTP` request/response, as the content size is as yet unknown. *Chunked encoding* has the benefit that it is not necessary to generate the full content before writing the header, as it allows streaming of content as chunks and explicitly signaling the end of the content, making the connection available for the next `HTTP` request/response.

Question 2
Thread Control

a. The program in the following figure has a bug. The thread is supposed to sleep for one second and then print a string. However, when we run it, nothing prints. Why?

=> The exit(0) function in main() just terminates all running thread including a new-born thread to sleep for one second and print a string as well.

b. You can fix this bug by replacing the exit function in the main with one of Pthreads function calls. Which one?

=> exit(0) → pthread_exit(0)

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *thread(void *vargp)
{
    sleep(1);
    printf("Hello world!\n");
    return NULL;
}

int main()
{
    pthread_t tid;
    pthread_create(&tid, NULL, thread, NULL);
    exit(0);
}
```