Socket Programming Project – Python HTTP Server, *Parallel Edition*

Name	Major	Checkpoints (50 pts)	Primary Objective (60 pts)	Secondary Objectives (40 pts)	Total Score (100 pts)

Checkpoints (50 points possible):
Checkpoints are graded as full credit *(25 points),* half credit *(13 points), or* no credit *(0 points)*

Description	Points	Score
Checkpoint 1: Server supports persistent connections, where a client can use one socket to sequentially request multiple files. Server also generates the required HTTP response headers, and supports varying the recv() size at the command line.	**25**	
Checkpoint 2: Server supports parallelism (thread or process based), whereby a client can use multiple sockets to request multiple files in parallel. Server also supports silent and verbose operation modes.	**25**	
Total (50 pts possible)		

Evaluation Rubric – Primary Objective (60 pts possible):
Web server can support concurrent requests from multiple clients simultaneously

Description	Points	Score
Server successfully uses threads or parallel processes to service multiple client sockets concurrently. Readme.txt file briefly describes parallelism choice.	**45**	
Siege benchmark configuration and results submitted for *original* web server and *new* parallel web server.	**15**	
Total (60 pts possible)		

Evaluation Rubric – Secondary Objectives (40 pts possible):

Description	Points	Score
Server supports HTTP persistent connections (socket stays open 30 seconds after a request, waiting for more requests)	**10**	
Server supports user-configurable recv() sizes via `--recv` command line argument	**10**	
Server gracefully shuts down upon receiving user CTRL-C	**5**	
Server supports HTTP HEAD requests	**5**	
Server provides the following HTTP response headers: Date, Server, Content-Length, Content-Type, Last-Modified, Expires	**5**	
Server provides verbose and silent modes of operation	**5**	
Total (40 pts possible)		

Comments: