Question 1

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/wait.h>
#include <string.h>
#include <stdbool.h>


//approved bash commands
bool validateCMD(char cmd[]) {
   if (strcmp(cmd, "/bin/mkdir") == 0)
      return true;
   if (strcmp(cmd, "/bin/ls") == 0)
      return true;
   if (strcmp(cmd, "/bin/cp") == 0)
      return true;
   if (strcmp(cmd, "/bin/mv") == 0)
      return true;
   if (strcmp(cmd, "/bin/vi") == 0)
      return true;
   else
      return false;
}
int main(int argc, char *argv[]) {
   pid_t pid;
   int ret = 1;
   int status;
   //extract command root
   //concat full bash command path '/bin/command'
```

```
char* cmd_raw = argv[1];
char* cmd = malloc(strlen(cmd_raw)+1+5);
strcpy(cmd, "/bin/");
strcat(cmd, cmd_raw);

if (validateCMD(cmd)) {
   //extract command parameters
   int PARMCOUNT = argc;
   //make sure arguments provided
   if (PARMCOUNT != 0) {
      char* parm[PARMCOUNT];
      //fill parmater array in complience with execv argument standard
      for (int index = 0; index < argc; index++)
         parm[index] = argv[index+1];
      //assign numm to end of argument list
      parm[PARMCOUNT-1] = NULL;
      //create child process to run command
      pid = fork();
      if (pid == -1) {
         //error occured
         printf("BASH-PROXY: child process couldn't be created.\n");
         exit(EXIT_FAILURE);
      } else if (pid == 0) {
         // child process created
         printf("BASH-PROXY: child process, pid = %u\n",getpid());
         //execute command
         execv(cmd,parm);
         exit(0);
      } else {
```

```
            // a positive number is returned for the pid of parent process
            printf("BASH-PROXY: parent process, pid = %u\n",getppid());
            // check execution and child process state
            if (waitpid(pid, &status, 0) > 0) {
               //successful execution
               if (WIFEXITED(status) && !WEXITSTATUS(status))
                  printf("BASH-PROXY: command execution successful.\n");
               else if (WIFEXITED(status) && WEXITSTATUS(status)) {
                  //execution failed
                  if (WEXITSTATUS(status) == 127) {
                     //execv failed
                     printf("BASH-PROXY: command failed.\n");
                  } else
                     printf("BASH-PROXY: command terminated normally, but returned a non-zero
status.\n");
               } else
                  printf("BASH-PROXY: program didn't terminate normally.\n");
            } else
               printf("BASH-PROXY: waitpid() failed.\n");


            exit(0);
         }
      } else
         printf("BASH-PROXY: no arguments provided.\n");
   } else
      printf("BASH-PROXY: command not supported.\n");


   return 0;
}
```

Question 2

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

// A normal C function that is executed as a thread
// when its name is specified in pthread_create()
void *myThreadFun(void *vargp)
{
            int max;
   //get max number of primes
   printf("Enter max prime number: ");
   scanf("%d",&max);
   //incremental loop to max prime estimte provided
   for (int number = 2; number < max; number++) {
      int isPrime = 1;
      //check divisibility
      for (int divisor = 2; divisor < number; divisor++)
         if ((number % divisor) == 0)
            isPrime = 0;

      if (isPrime == 1)
         printf("%d ", number);
   }

  return 0;
}
```

```
int main()
{
	pthread_t thread_id;
	printf("Thread Created....\n\n");
	pthread_create(&thread_id, NULL, myThreadFun, NULL);
	pthread_join(thread_id, NULL);
	printf("\n\nThread Finished.\n");
	exit(0);
}
```