**Assignment 5**

1.

Server code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

unsigned long long factorial(int n)

{

   unsigned long long result = 1;

   for (int i = 2; i <= n; i++)

   {

      result \*= i;

   }

   return result;

}

int main()

{

   int server\_fd, new\_socket;

   struct sockaddr\_in address;

   int addrlen = sizeof(address);

   int num;

   unsigned long long fact\_result;

   server\_fd = socket(AF\_INET, SOCK\_STREAM, 0);

   address.sin\_family = AF\_INET;

   address.sin\_addr.s\_addr = INADDR\_ANY;

   address.sin\_port = htons(8080);

   bind(server\_fd, (struct sockaddr \*)&address, sizeof(address));

   listen(server\_fd, 3);

   new\_socket = accept(server\_fd, (struct sockaddr \*)&address, (socklen\_t \*)&addrlen);

   read(new\_socket, &num, sizeof(num));

   fact\_result = factorial(num);

   write(new\_socket, &fact\_result, sizeof(fact\_result));

   close(new\_socket);

   close(server\_fd);

   return 0;

}

Client Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

int main()

{

   int sock = 0;

   struct sockaddr\_in serv\_addr;

   int num;

   unsigned long long fact\_result;

   sock = socket(AF\_INET, SOCK\_STREAM, 0);

   serv\_addr.sin\_family = AF\_INET;

   serv\_addr.sin\_port = htons(8080);

   inet\_pton(AF\_INET, "127.0.0.1", &serv\_addr.sin\_addr);

   connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr));

   printf("Enter an integer: ");

   scanf("%d", &num);

   write(sock, &num, sizeof(num));

   read(sock, &fact\_result, sizeof(fact\_result));

   printf("Factorial of %d is %llu\n", num, fact\_result);

   close(sock);

   return 0;

}

Output:



2.

Mapper.c

#include <stdio.h>

#include <stdlib.h>

#include <dirent.h>

#include <sys/stat.h>

#include <pwd.h>

#include <unistd.h>

int main()

{

   struct dirent \*entry;

   struct stat file\_stat;

   DIR \*dp = opendir(".");

   if (dp == NULL)

   {

      perror("opendir");

      return EXIT\_FAILURE;

   }

   while ((entry = readdir(dp)))

   {

      if (entry->d\_type == DT\_REG)

      {

         if (stat(entry->d\_name, &file\_stat) == 0)

         {

            struct passwd \*pw = getpwuid(file\_stat.st\_uid);

            if (pw)

            {

               printf("%ld\t%s\n", file\_stat.st\_size, pw->pw\_name);

            }

         }

      }

   }

   closedir(dp);

   return EXIT\_SUCCESS;

}

Reducer.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_LINE\_LENGTH 256

int main()

{

   char line[MAX\_LINE\_LENGTH];

   long max\_size = 0;

   char owners[MAX\_LINE\_LENGTH][MAX\_LINE\_LENGTH];

   int owner\_count = 0;

   long file\_size;

   char owner[MAX\_LINE\_LENGTH];

   while (fgets(line, sizeof(line), stdin))

   {

      sscanf(line, "%ld\t%s", &file\_size, owner);

      if (file\_size > max\_size)

      {

         max\_size = file\_size;

         owner\_count = 0;

         strcpy(owners[owner\_count++], owner);

      }

      else if (file\_size == max\_size)

      {

         int found = 0;

         for (int i = 0; i < owner\_count; i++)

         {

            if (strcmp(owners[i], owner) == 0)

            {

               found = 1;

               break;

            }

         }

         if (!found)

         {

            strcpy(owners[owner\_count++], owner);

         }

      }

   }

   for (int i = 0; i < owner\_count; i++)

   {

      printf("%s\n", owners[i]);

   }

   return EXIT\_SUCCESS;

}

Output:

