LAB-6 Programs on Strings in CUDA

In a multithreaded scenario, the issue of data inconsistency will arise, if multiple threads modify a single shared memory variable. To overcome this atomic functions need to be used. List of atomic functions, their syntax and explanation is provided below.

atomicAdd():

```
int atomicAdd (int* address, int val);

unsigned int atomicAdd(unsigned int* address,  unsigned int val);

float atomicAdd(float* address, float val);

double atomicAdd(double* address, double val);
```

Reads the 16-bit, 32-bit or 64-bit word old located at the address address in global or shared memory, computes (old + val), and stores the result back to memory at the same address. These three operations are performed in one atomic transaction. The function returns old.

Solved Example:

A CUDA program which takes a string as input and determines the number of occurences of a character 'a' in the string. This program uses atomicAdd() function.

```
#include "cuda_runtime.h"
#include "device_launch_parameters.h"

#include <stdio.h>
#include <stdlib.h>
#include<string.h>
#include <conio.h>
#define N 1024

__global__ void CUDACount(char* A, unsigned int *d_count){
        int i = threadIdx.x;
                if(A[i]=='a')
                        atomicAdd(d_count,1);
    }
int main()  {
char A[N];
char *d_A;
unsigned int *count=0,*d_count,*result;
printf("Enter a string");
gets(A);
cudaEvent_t start, stop;
cudaEventCreate(&start);
cudaEventCreate(&stop);
cudaEventRecord(start, 0);
cudaMalloc((void**)&d_A, strlen(A)*sizeof(char));
cudaMalloc((void **)&d_count,sizeof(unsigned int));
cudaMemcpy(d_A, A, strlen(A)*sizeof(char), cudaMemcpyHostToDevice);
cudaMemcpy(d_count,count,sizeof(unsigned int),cudaMemcpyHostToDevice);
```

```
cudaError_t error =cudaGetLastError();
if (error != cudaSuccess) {
                    printf("CUDA  Error1:  %s\n",  cudaGetErrorString(error));
        }
CUDACount<<<1,strlen(A)>>>(d_A,d_count);
error =cudaGetLastError();
if (error != cudaSuccess) {
                    printf("CUDA  Error2:  %s\n",  cudaGetErrorString(error));

        }
cudaEventRecord(stop, 0);
cudaEventSynchronize(stop);
float elapsedTime;
cudaEventElapsedTime(&elapsedTime, start, stop);
cudaMemcpy(result, d_count, sizeof(unsigned int), cudaMemcpyDeviceToHost);
printf("Total occurences of a=%u",result);
printf("Time Taken=%f",elapsedTime);
cudaFree(d_A);
cudaFree(d_count);
printf("\n");
getch();
return 0;
}
```

Lab Exercises:

1. Write a program in CUDA to count the number of times a given word is repeated in a sentence. (Use Atomic function)

2.Write a CUDA program that reads a string ***S*** and produces the string ***RS*** as follows:
Input string ***S***: PCAP Output string ***RS***: PCAP**PCA**PC**P**
Note: Each work item copies required number of characters from *S* in *RS*.

Additional Exercises:
1) Write a CUDA program which reads a string consisting of N words and reverse each word of it in parallel.

2) Write a CUDA program that takes a string *Sin* as input and one integer value N and produces an output string , *Sout,* in parallel by concatenating input string *Sin*, N times as shown below.
Input: Sin = "Hello" N = 3
Ouput: Sout = "HelloHelloHello"
Note: Every thread copies the same character from the Input string S, N times to the required position.
3)Write a CUDA program which reads a string *Sin* and produces an output string T as shown below.
Input: Sin: "Hai"
Ouput: *T*: "Haaiii "

Note:Every thread stores a character from input string *Sin*, required number of times into output string *T*.