PPL LAB-6 EXERCISES

📅 Date @February 20, 2026

TITLE : PROGRAMS ON STRINGS IN CUDA

NAME : DEVADATHAN N R

ROLL NO : 04

SECTION : CSE A1

REG NO : 230905010

Question 1:

Write a CUDA program to perform 1D convolution on an input array
using a mask array and produce the output array.

Code:

```
#include <stdio.h>
#include <cuda.h>

#define N 7
#define M 5

__global__ void conv1D(float *in, float *mask, float *out, int n)
{
    int tid = blockIdx.x * blockDim.x + threadIdx.x;

    float sum = 0;
    int start = tid - (M / 2);

    for (int i = 0; i < M; i++)
    {
```

```c
            if (start + i >= 0 && start + i < n)
            {
                sum += in[start + i] * mask[i];
            }
        }

    out[tid] = sum;
}

int main()
{
    float h_in[N]  = {2, 4, 6, 8, 10, 12, 14};
    float h_mask[M] = {1, 0, -1, 0, 1};
    float h_out[N];

    float *d_in, *d_mask, *d_out;

    cudaMalloc(&d_in,  N * sizeof(float));
    cudaMalloc(&d_mask, M * sizeof(float));
    cudaMalloc(&d_out, N * sizeof(float));

    cudaMemcpy(d_in, h_in, N * sizeof(float), cudaMemcpyHostT
oDevice);
    cudaMemcpy(d_mask, h_mask, M * sizeof(float), cudaMemcpyH
ostToDevice);

    conv1D<<<1, N>>>(d_in, d_mask, d_out, N);

    cudaMemcpy(h_out, d_out, N * sizeof(float), cudaMemcpyDev
iceToHost);

    printf("Output Array:\n");
    for (int i = 0; i < N; i++)
        printf("%f ", h_out[i]);

    printf("\n");
```

```
        cudaFree(d_in);
        cudaFree(d_mask);
        cudaFree(d_out);

        return 0;
    }
```

Output:

```
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ nvcc l6q1.cu -o l6q1
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ ./l6q1
Output Array:
4.000000 4.000000 6.000000 8.000000 10.000000 -4.000000 -4.000000
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$
```

Question 2:

CUDA program to perform parallel selection sort.

Code:

```
#include "cuda_runtime.h"
#include "device_launch_parameters.h"
#include <stdio.h>

__global__ void selSort(int *d_in, int *d_out, int n)
{
    int tid = blockIdx.x * blockDim.x + threadIdx.x;

    int rank = 0;

    for (int i = 0; i < n; i++)
    {
        if ((d_in[i] < d_in[tid]) ||
```

```
                (d_in[i] == d_in[tid] && i > tid))
        {
                rank++;
        }
    }

    d_out[rank] = d_in[tid];
}

int main()
{
    int n;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    int h_in[n];
    int h_out[n];

    int *d_in, *d_out;

    cudaMalloc((void **)&d_in, n * sizeof(int));
    cudaMalloc((void **)&d_out, n * sizeof(int));

    printf("Enter the elements: ");
    for (int i = 0; i < n; i++)
        scanf("%d", &h_in[i]);

    cudaMemcpy(d_in, h_in, n * sizeof(int), cudaMemcpyHostToDevice);

    selSort<<<1, n>>>(d_in, d_out, n);

    cudaMemcpy(h_out, d_out, n * sizeof(int), cudaMemcpyDeviceToHost);
```

```
        printf("Sorted array: ");
        for (int i = 0; i < n; i++)
            printf("%d ", h_out[i]);

        printf("\n");
        cudaFree(d_in);
        cudaFree(d_out);

        return 0;
}
```

Output:

```
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ nvcc l6q2.cu -o l6q2
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ ./l6q2
Enter number of elements: 6
Enter the elements: 8 4 6 2 9 1
Sorted array: 1 2 4 6 8 9
```

Question 3:

CUDA program to perform Odd-Even Transposition Sort in parallel.

Code:

```
#include <stdio.h>
#include <math.h>
#include "cuda_runtime.h"
#include "device_launch_parameters.h"

__global__ void oddEvenSort(int *d_arr, int n, int phase)
{
    int tid = threadIdx.x + blockDim.x * blockIdx.x;
```

```
        int idx = (phase % 2 == 0) ? (2 * tid) : (2 * tid + 1);

    if (idx + 1 < n)
    {
        if (d_arr[idx] > d_arr[idx + 1])
        {
            int temp = d_arr[idx];
            d_arr[idx] = d_arr[idx + 1];
            d_arr[idx + 1] = temp;
        }
    }
}

int main()
{
    int n;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    int h_arr[n];
    int *d_arr;

    printf("Enter the elements: ");
    for (int i = 0; i < n; i++)
        scanf("%d", &h_arr[i]);

    cudaMalloc((void **)&d_arr, n * sizeof(int));
    cudaMemcpy(d_arr, h_arr, n * sizeof(int), cudaMemcpyHostT
oDevice);

    for (int p = 0; p < n; p++)
    {
        oddEvenSort<<<ceil((float)n / 512), 512>>>(d_arr, n,
p);
        cudaDeviceSynchronize();
```

```
        }

    cudaMemcpy(h_arr, d_arr, n * sizeof(int), cudaMemcpyDevic
eToHost);

    printf("Sorted array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", h_arr[i]);

    printf("\n");
    cudaFree(d_arr);

    return 0;
}
```

Output:

```
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ nvcc l6q3.cu -o l6q3
^[[ASTUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$ ./l6q3
Enter number of elements: 8
Enter the elements: 9 4 7 2 6 1 8 3
Sorted array: 1 2 3 4 6 7 8 9
STUDENT@MIT-ICT-LAB5-4:~/Desktop/230905010/06_Lab$
```