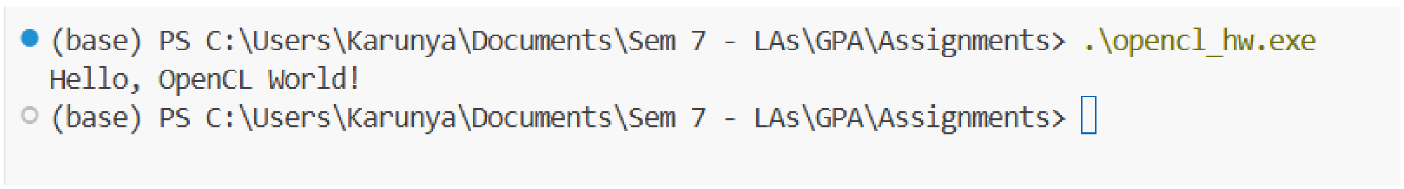
ASSIGNMENT - IV

```
#include <stdio.h>
#include <stdlib.h>
#include <CL/cl.h>

int main(void) {
  // Kernel source
  const char *source_str =
    "__kernel void hello_world(__global char* output) {"
    "   const __constant char message[] = \"Hello, OpenCL World!\";"
    "   int i;"
    "   for (i = 0; message[i] != '\\0'; i++)"
    "       output[i] = message[i];"
    "   output[i] = '\\0';"
    "}";

  // Platform/device setup
  cl_platform_id platform_id = NULL;
  cl_device_id device_id = NULL;
  cl_uint ret_num_devices, ret_num_platforms;
  cl_int ret = clGetPlatformIDs(1, &platform_id, &ret_num_platforms);
  ret = clGetDeviceIDs(platform_id, CL_DEVICE_TYPE_DEFAULT, 1, &device_id, &ret_num_devices);

  // Context + queue
  cl_context context = clCreateContext(NULL, 1, &device_id, NULL, NULL, &ret);
  cl_command_queue command_queue = clCreateCommandQueue(context, device_id, 0, &ret);

  // Output buffer
  cl_mem output_mem_obj = clCreateBuffer(context, CL_MEM_WRITE_ONLY, 1024, NULL, &ret);

  // Program and kernel
  cl_program program = clCreateProgramWithSource(context, 1, &source_str, NULL, &ret);
  ret = clBuildProgram(program, 1, &device_id, NULL, NULL, NULL);
```

```
    if (ret != CL_SUCCESS) {
        size_t log_size;
        clGetProgramBuildInfo(program, device_id, CL_PROGRAM_BUILD_LOG, 0, NULL, &log_size);
        char *log = (char *)malloc(log_size);
        clGetProgramBuildInfo(program, device_id, CL_PROGRAM_BUILD_LOG, log_size, log, NULL);
        printf("Build error:\n%s\n", log);
        free(log);
        return 1;
    }

    cl_kernel kernel = clCreateKernel(program, "hello_world", &ret);
    ret = clSetKernelArg(kernel, 0, sizeof(cl_mem), (void *)&output_mem_obj);

    // Execute
    size_t global_item_size = 1;
    size_t local_item_size = 1;
    ret = clEnqueueNDRangeKernel(command_queue, kernel, 1, NULL, &global_item_size,
&local_item_size, 0, NULL, NULL);
    // Read back
    char output[1024];
    ret = clEnqueueReadBuffer(command_queue, output_mem_obj, CL_TRUE, 0, 1024, output, 0, NULL,
NULL);
    printf("%s\n", output);
    // Cleanup
    clFlush(command_queue);
    clFinish(command_queue);
    clReleaseKernel(kernel);
    clReleaseProgram(program);
    clReleaseMemObject(output_mem_obj);
    clReleaseCommandQueue(command_queue);
    clReleaseContext(context);
    return 0;
}
```

OUTPUT



```
● (base) PS C:\Users\Karunya\Documents\Sem 7 - LAs\GPA\Assignments> .\opencl_hw.exe
  Hello, OpenCL World!
○ (base) PS C:\Users\Karunya\Documents\Sem 7 - LAs\GPA\Assignments> 
```

Figure 1: Graphics Program Using OpenGL and CUDA.