i. Write a SV program for Memory Access & Writeback Stage

MEMORY ACCESS :

```
module memstage #(parameter N=32, M=1024) (
   input logic [N-1:0] alu_out, data_in,
   input logic mem_read, mem_write,
    output logic [N-1:0] data_out
);
logic [N-1:0] mem [0:M-1];
always_comb
        begin
        if( mem_read==1 && mem_write==0)
                data_out= mem[alu_out];
        else if (mem_read==0 && mem_write==1) begin
                mem[alu_out]= data_in;
                data_out= 32'b0;
                end
        else
                data_out= 32'b0;
        end
endmodule
```

WRITEBACK STAGE:

```
module write_back1 #(parameter N=32)(
        input logic [N-1:0] alu_result,
        input logic [N-1:0] data_out,
        input logic mem_reg,
        output logic [N-1:0] wr_data
        );
        //memory_ME memory (.alu_result(alu_result),.data_in(data_in),.mem_read(mem_read),.mem_write(mem_write),.data_out(data_out));
        mux_1 wb_mux (alu_result,data_out,mem_reg,wr_data);
```

```
          endmodule
```

ii.Write a SV program for Data memory

```
module  data_mem #(parameter N=32) (
   input logic [N-1:0] alu_out, data_in,
   input logic mem_read, mem_write,
   output logic [N-1:0] data_out
);

memstage g1(alu_out, data_in,mem_read, mem_write,data_out);
endmodule
```