ASSIGNMENT:

A function with the prototype `int ISRDIsampling();` is used in an ISR (Interrupt Service

Routine) to poll the condition of 8 GPIO pins at 100Hz, a variable `g\_ReadDIpinSts` defined

globally contains the status of each of these GPIO pins at any given time, this variable is

updated elsewhere.

A variable `g\_AppDIpinSts` defined in global scope is to be updated by function `int

ISRDIsampling();` to showcase the actual status of each pin.

Complete function: `int ISRDIsampling();` such that each GPIO state is preserved in variable

`g\_AppDIpinSts` for at least 10 ISR calls before it is reset by the ISR.

CODE:

#include <stdio.h>

#include <stdint.h>

#include <unistd.h>

// defining varaibles globally

uint8\_t g\_ReadDIpinSts = 0b00100000;

uint8\_t g\_AppDIpinSts = 0b00000000;

int count = 0;

int ISR\_DIsampling()

{

// Updating g\_AppDIpinSts only after 10 ISR calls

if (count >= 10) {

g\_AppDIpinSts = g\_ReadDIpinSts;

count = 0; // Resetting the counter

}

count++; // Incrementing the counter

return 0; // Returning zero

}

//Main program

int main()

{

while (1) {

ISR\_DIsampling();// Calling the ISR\_DIsampling function

printf("g\_AppDIpinSts: 0x%02X\n", g\_AppDIpinSts);

usleep(10000);//delay of 10 milliseconds

}

return 0;//returning zero to the main program

}