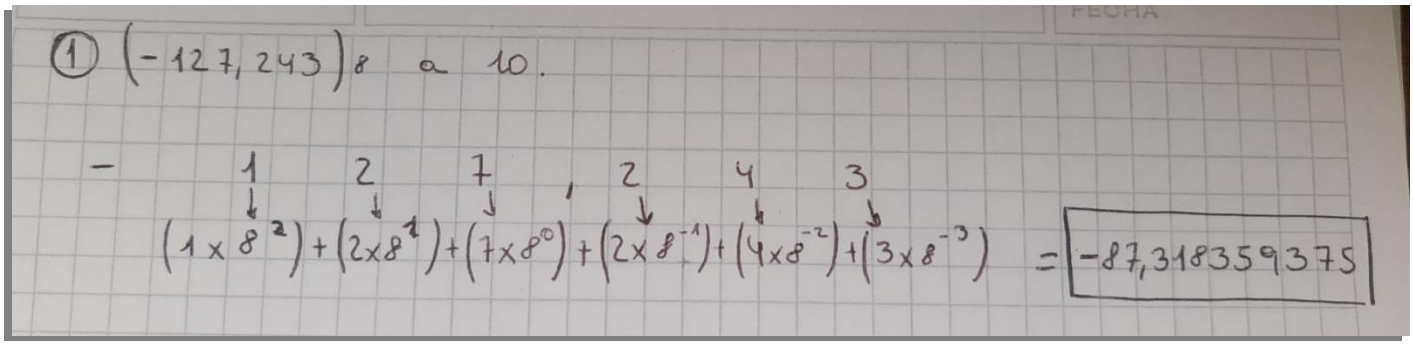
UTN - INSPT
2.601 - Sistemas de Computación I
Parcial 2021

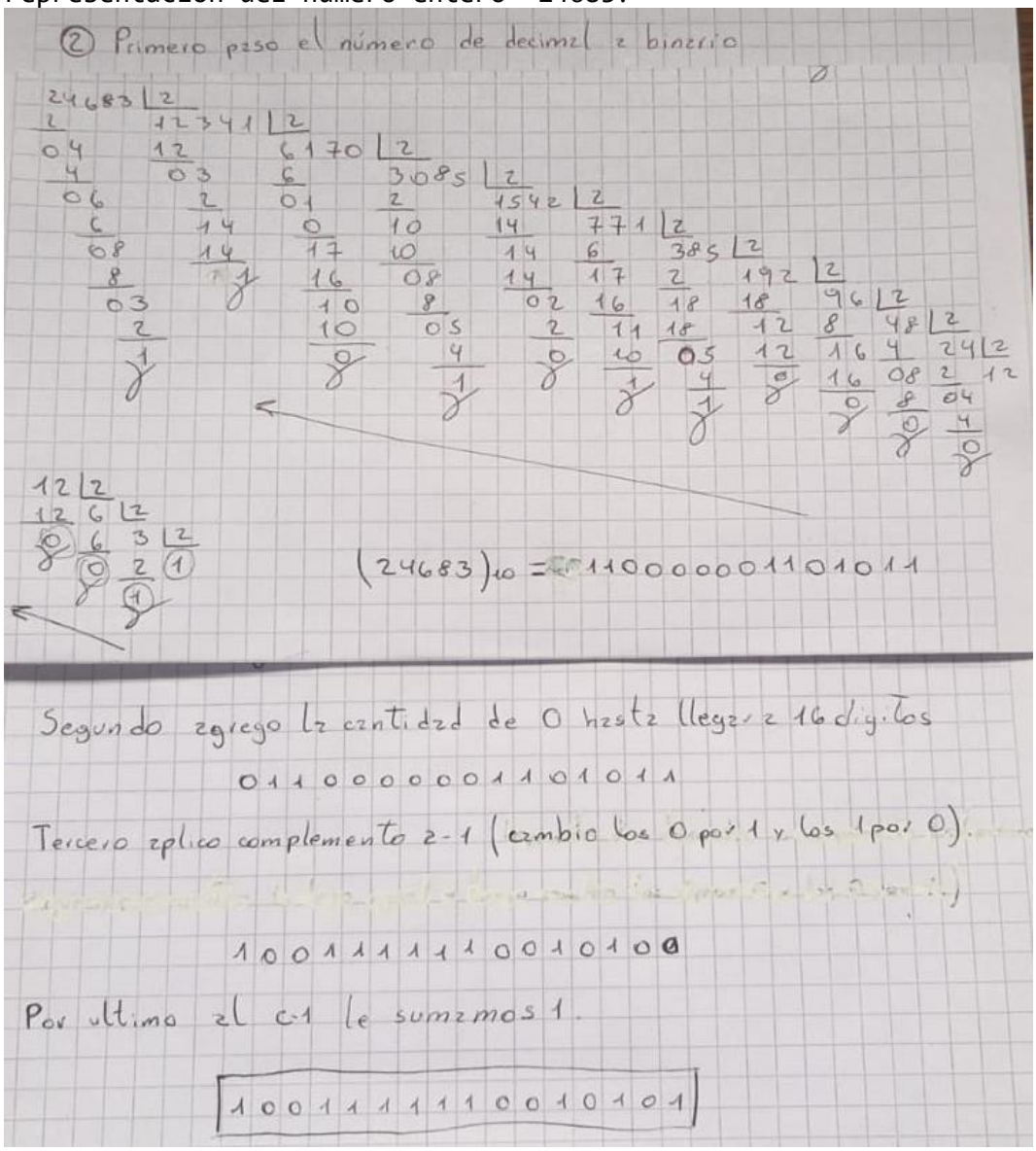
Apellido(s), Nombre(s): Alonso Lucio

Plantee y resuelva las siguientes cuestiones. Deberá insertar debajo de cada una de las preguntas 1 a 9, claramente legibles de su puño y letra, los pasos seguidos detalladamente y la solución (pueden ser imágenes escaneadas o fotos). La explicación del punto 10 deberá ser de puño y letra y la implementación deberá estar subida en GitLab o GitHub, con el docente invitado a verla como reporter.
El parcial resuelto deberá enviarse, en un archivo PDF, hasta el 09/11/2021 a: corsi@mail.com

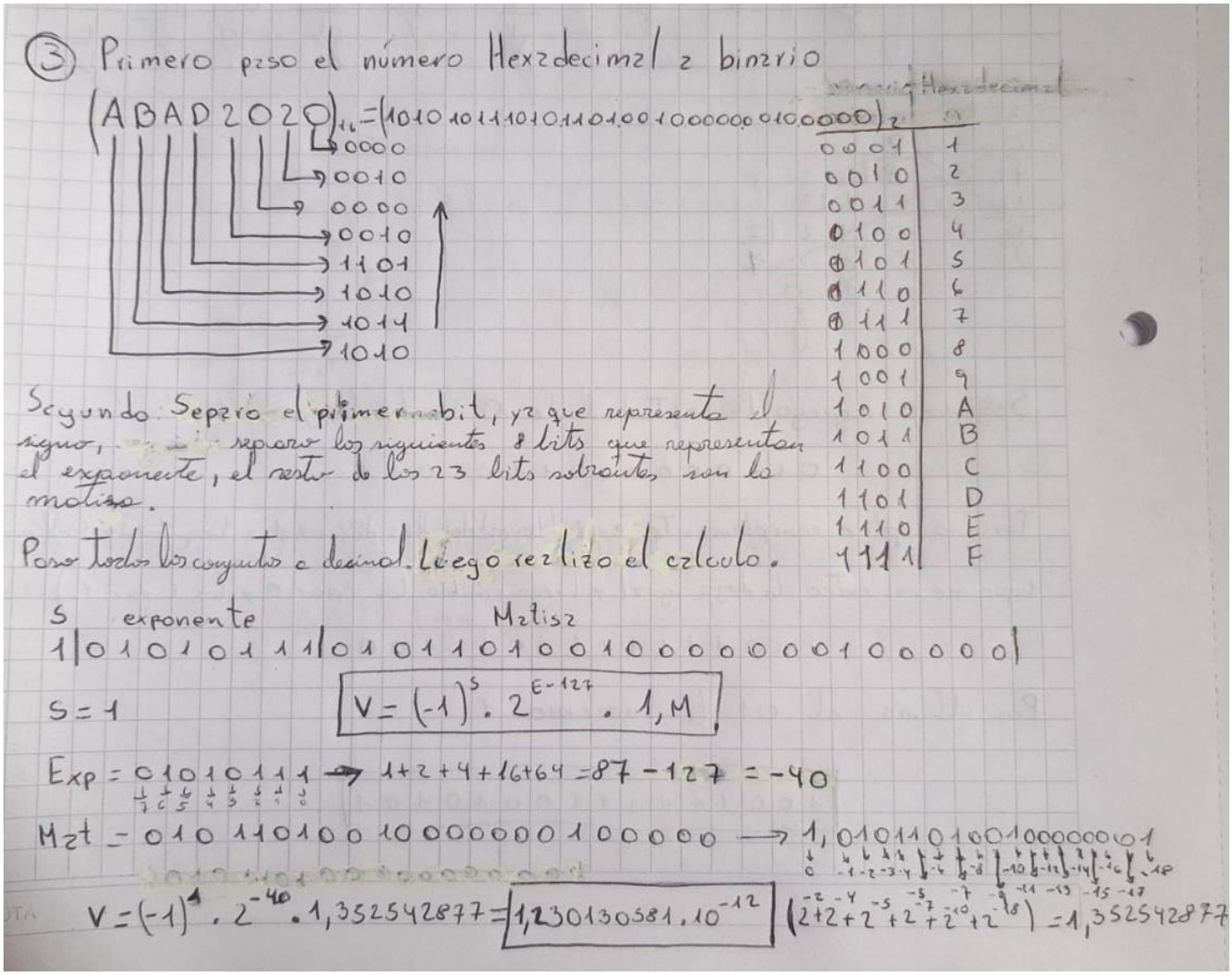
1) ¿Cuál es, en decimal, la representación del número octal -127,243?



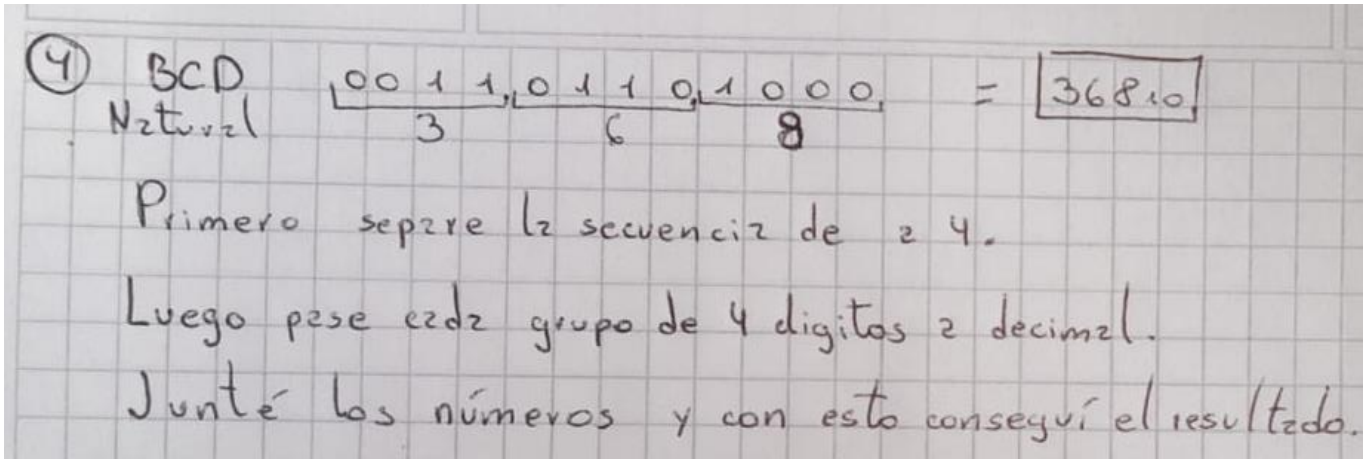
2) ¿Cuáles son, al usar Complemento a 2 (C-2), los 16 bits de la representación del número entero -24683?



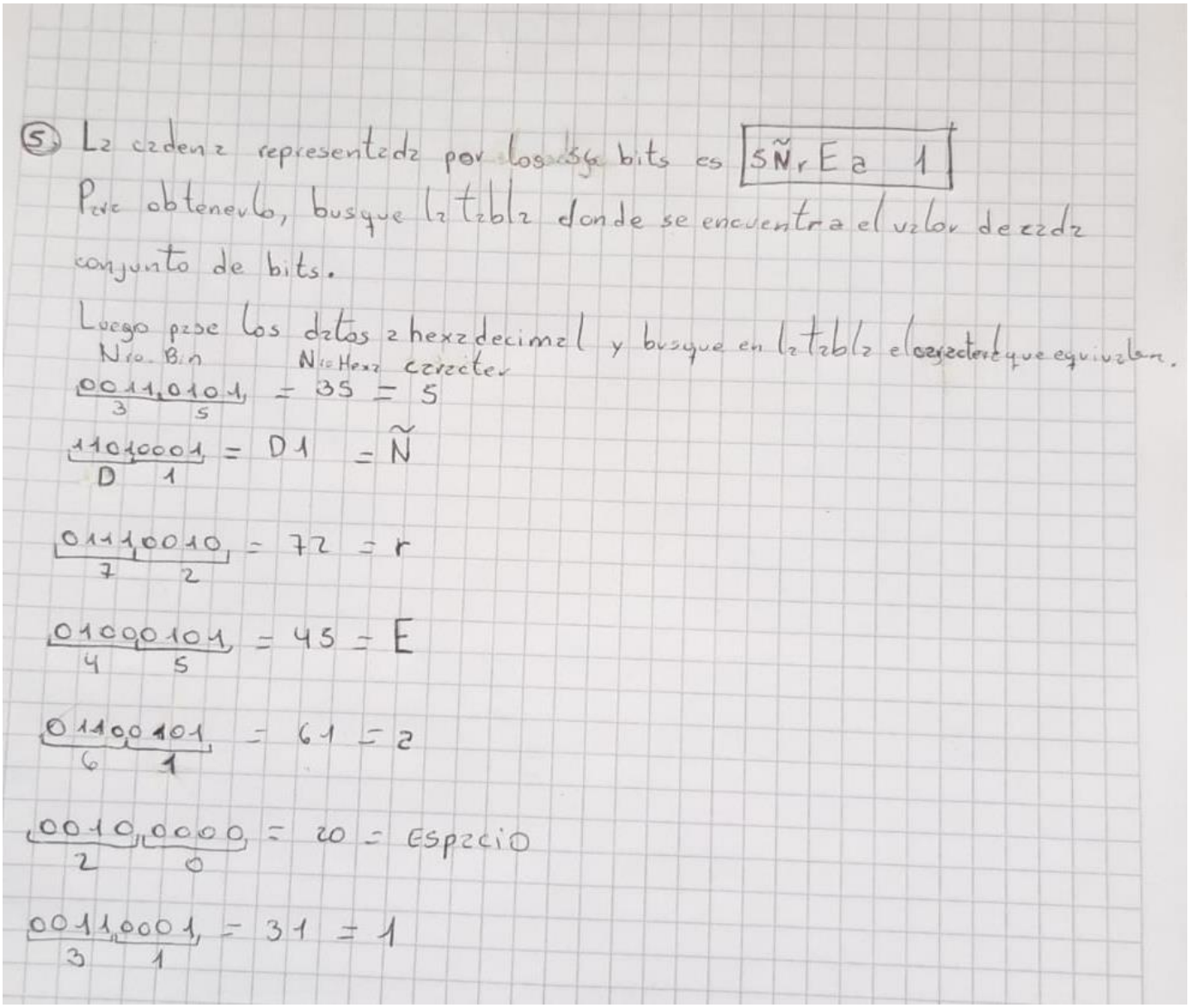
3) ¿Cuál es, al usar IEEE 754 (single), el número decimal representado por los 32 bits expresados mediante los 8 dígitos hexadecimales ABAD2020?



4) ¿Cuál es el número que, al ser representado usando el Código 8421 (BCD natural), da como resultado la secuencia binaria 001101101000?



5) ¿Cuál es, al usar el Código ISO/IEC 8859-1, la cadena representada por los 56 bits 00110101 11010001 01110010 01000101 01100001 00100000 00110001?



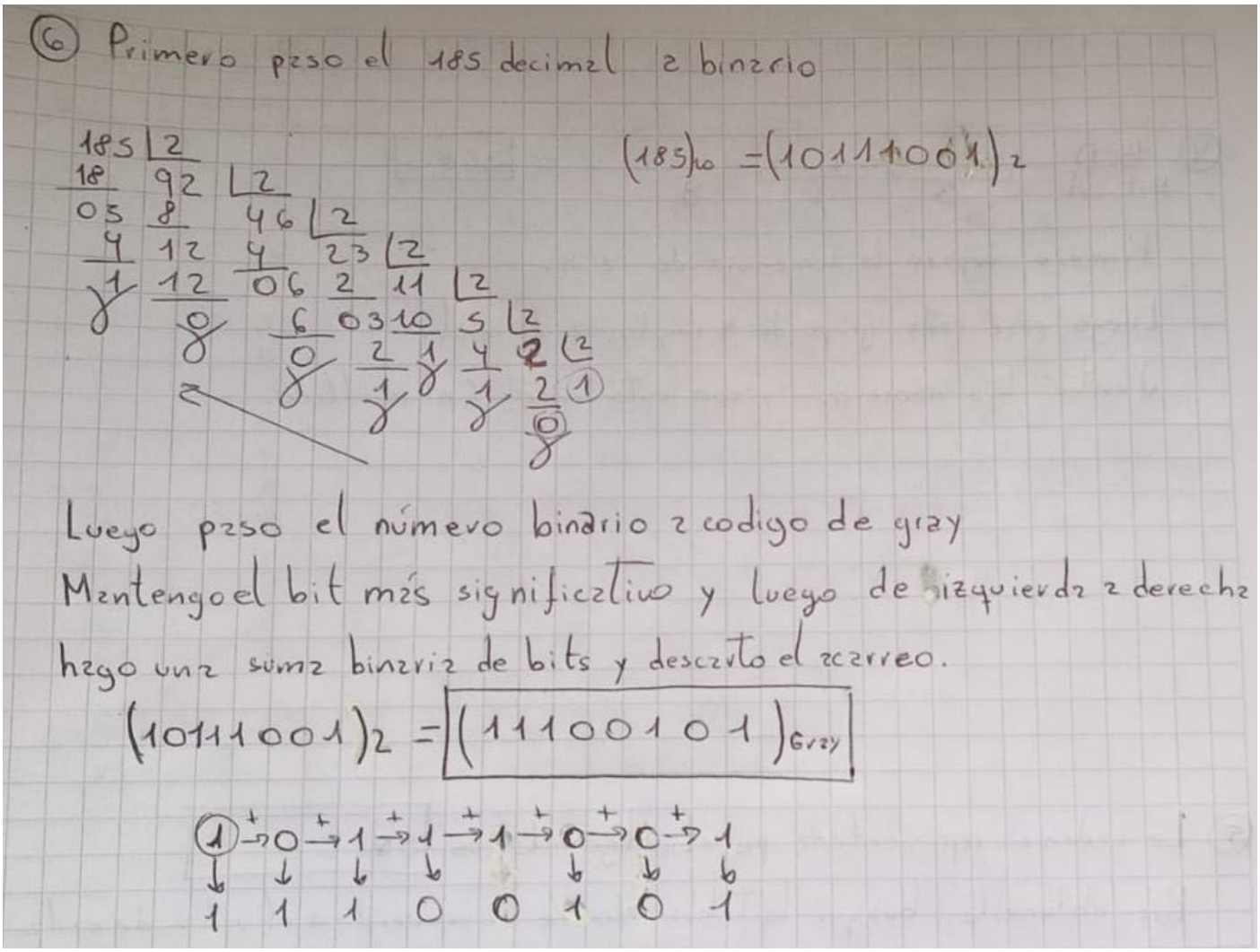
(5) La cadena representada por los 56 bits es 5ÑrEa 1

Para obtenerlo, busque la tabla donde se encuentra el valor de cada conjunto de bits.

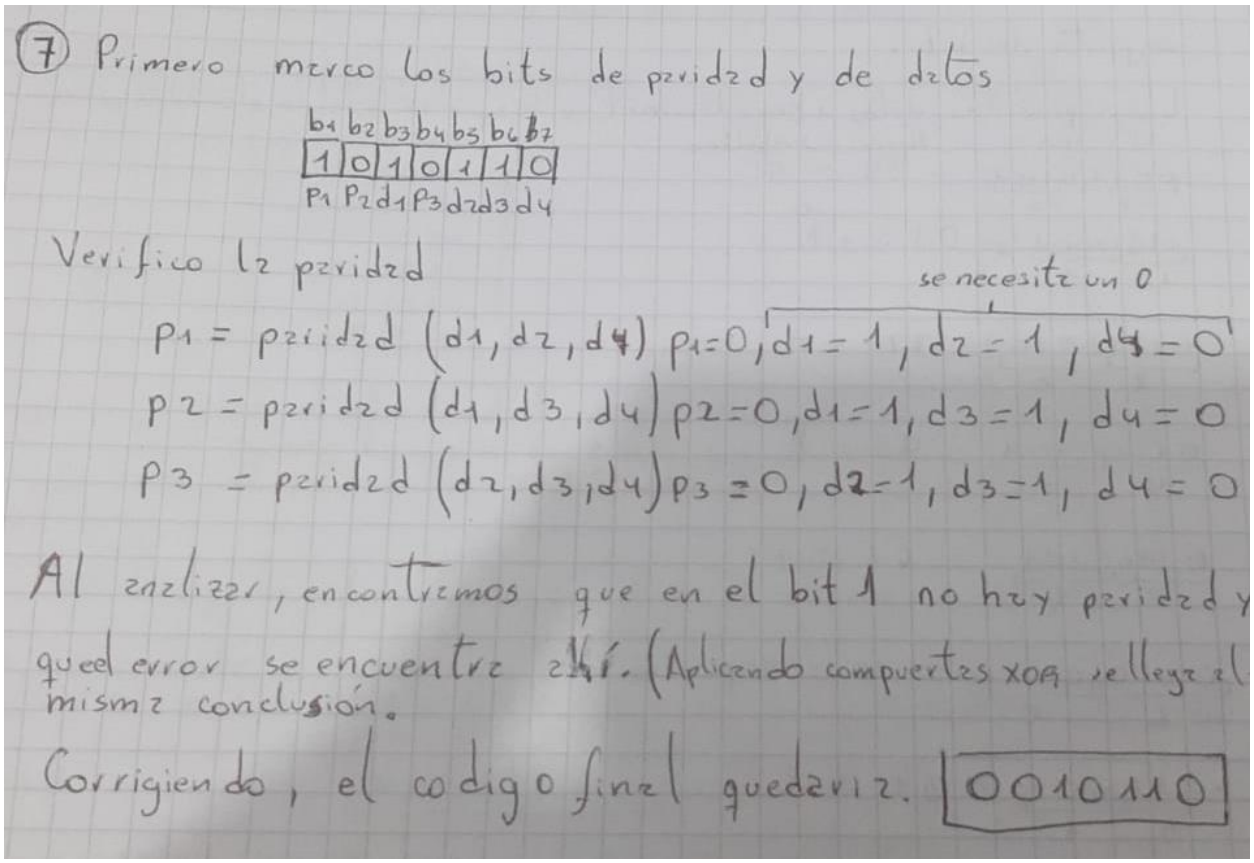
Luego pase los datos a hexadecimal y busque en la tabla el caracter que equivalen.

Num. Bin	Num Hex	caracter
00110101	3 5 = 35	5
11010001	D 1 = D1	Ñ
01110010	7 2 = 72	r
01000101	4 5 = 45	E
01100001	6 1 = 61	a
00100000	2 0 = 20	Espacio
00110001	3 1 = 31	1

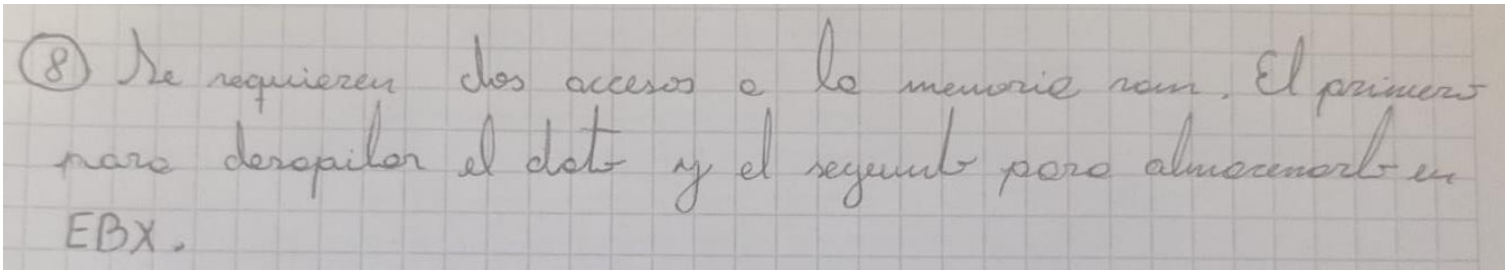
6) ¿Cuáles son los 8 bits con los que se representa el número decimal 185 al usar el código binario reflejado de Gray?



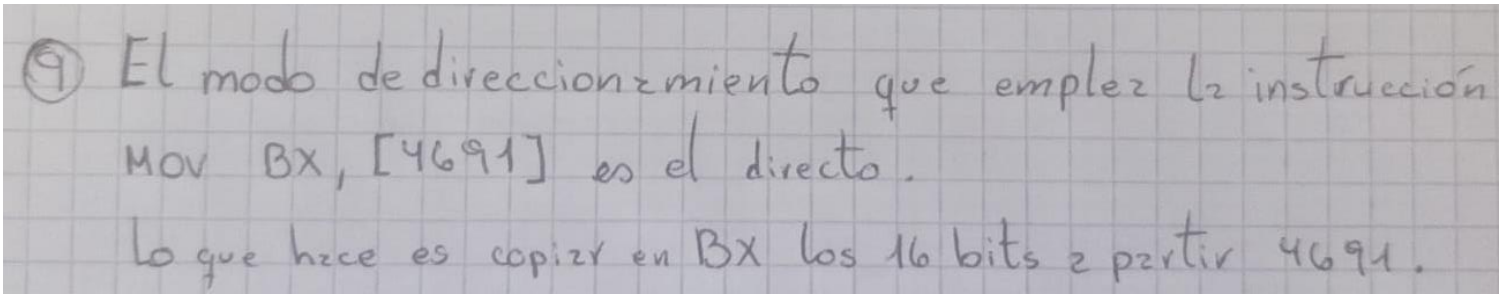
7) ¿Cuáles fueron los 7 bits enviados usando el código de Hamming (7, 4), si se recibió la siguiente secuencia binaria con (como máximo) un bit erróneo? 1010110



8) ¿Cuántos accesos a RAM se requieren para ejecutar la instrucción POP EBX? Explique y justifique su respuesta



9) ¿Qué modo de direccionamiento utiliza la instrucción MOV BX, [4691]? Explíquelo basándose en la codificación de la instrucción cuando está alojada en la RAM.



10) Explique e implemente las modificaciones requeridas por su compilador de PL/0 para que, a partir del código fuente que se muestra a continuación, genere un archivo PE-32 (para Windows) o ELF (para GNU/Linux) que corra según los ejemplos.

```
const MIN = -10000, max=10000;

var A,B,C,W,X,Y,Z;

procedure MOSTRAR;
begin
  IF Y = 0 THEN WRITELN ('HAY UNA RAIZ EN X=', X);
  W := 1
end;

begin
  WRITELN;
  WRITELN ('PROGRAMA PARA BUSCAR RAICES ENTERAS DE ECUACIONES CUADRATICAS');
  WRITELN;
  WRITELN ('   2');
  WRITELN ('A X + B X + C = 0');
  WRITELN;
  write ('Ingrese A: '); readLn (A);
  write ('Ingrese B: '); readLn (b);
  write ('Ingrese C: '); readLn (c);
  WRITELN;
  x := min;
  W := 0;
  while x <= max do
    begin
       z := x - 1;
       y := a * sqr(z+1) + B * x + c;
       if y = 0 then call MOSTRAR;
       x := x + 1
    end;
  if W = 0 then writeln ('NO HAY RAICES ENTERAS ENTRE ', MIN, ' Y ', MAX)
end.
```

Ejemplos:

$2\ x^2 - 6\ x + 4 = 0$
Tiene 2 raíces enteras (x = 1; x = 2)

$3\ x^2 = 0$
Tiene una raíz entera (x = 0)

$x^2 - 1 = 0$
Tiene 2 raíces enteras (x = -1; x = 1)

$2\ x^2 + 2\ x + 2 = 0$
No tiene raíces enteras

(10) Las modificaciones implementadas en el compilador fueron las siguientes:

1° Se agregó en la función BLOQUE (dentro del if que detecta un simbolo CONST), que luego de leer un IGUAL (=), puede leerse a continuación un NEGATIVO (-) y un Número. Si no entra un negativo, se busca un número.

Código:

```
if ((*simbolo)==MENOS)
  { cadena_t simboloNeg;
    strcpy (simboloNeg, cadenaSimbolo);
    escanear (f, restante, simbolo, cadenaSimbolo, contadorRenglon);
    if ((*simbolo)==NÚMERO)
    { strcat (simboloNeg, cadenaSimbolo);
      strcpy (cadenaSimbolo, simboloNeg);
      tablaSimb [base+desplazamiento].valor = atoi (cadenaSimbolo);
      escanear (f, restante, simbolo, cadenaSimbolo, contadoRenglon);
    } else mensajeErr (f, s, cadenaSimbolo);
  }
```

2° Luego se modificó la función búsqueda, ya que esta diferenciaba entre minúsculas y mayúsculas. Esto provocaba que a la hora de querer leer una variable que había sido escrita con mayúsculas en su declaración, no se la lograra reconocer como tal si más adelante en el código se la escribía en minúsculas.

Para arreglar esto, se transformó, en la comparación de los símbolos, que ambos estén en mayúsculas usando strupr.

Código: i): if (strcmp(strupr(nombreSimb), strupr(tablaSimb[i].nombre))==0)
{ break; }

3° Por último, se agregó el SQR (para realizar la potencia) al código.
- Primero se colocó dentro del typedef enum el SQR.
- Segundo, dentro de la función factor, se agregó un case SQR, el cual es el encargado de realizar la potencia. Detecta la apertura de un paréntesis, espera una expresión, luego hace los pasajes de memoria correspondientes para realizar la potencia del número y por último, espera un cierre de paréntesis.

Código: case SQR:

```
escanear(f, resto, simbolo, cadenaSimb, contadorRenglon);
if((*simbolo) == ABREPARENTESIS) {
 escanear(f, resto, ....);
 expresión(f, resto, simbolo, cadenaSimb, contadorRenglon,
                tablaSimb, base, desplazamiento, topeMemoria, memoria);
 cargarByte(0x5B, memoria, topeMemoria);  // POP EBX
 cargarByte(0xB8, memoria, topeMemoria);  // MOV EAX X
 cargarInt(0x00, memoria, topeMemoria);
 cargarByte(    , memoria, topeMemoria);  // ADD EAX,EBX
 cargarByte(    , memoria, topeMemoria);
 cargarByte(    , memoria, topeMemoria);  // IMUL EBX
 cargarByte(    , memoria, topeMemoria);
 cargarByte(    , memoria, topeMemoria);  // push EAX
 if((*simbolo) == CIERRAPARENTESIS) {
   escanear(....);
 } else....
```

- En la función ... se agrego el simbolo SQR.

Código: ...else if (strcmp(cadena Aux, "SQR") == 0)
{ *simbolo = SQR; }

- Y en la función imprimirSimbolo se agregó un case SQR

Código: case SQR:
printf("SQR\n");
break;