Question 2 // The small blue number is for carryover / borrow.

ⓐ
```
  1011
+ 0011
------
  1110   (= 14₁₀)
```

Decimal: $(11)_{10}$ (8+2+1)

$+ (3)_{10}$ (2+1)

$(14)_{10}$ (8+4+2)

→ Correct result, fit k = 4 for the given rep.

ⓑ
```
  010100
+ 011101
--------
  110001
```
$= (32+16+1)$

$= 49_{10}$

Decimal: $16+4 = 20_{(10)}$

$+ \; 16+4+8+1 = 29_{(10)}$

$49_{(10)}$

→ Correct result, fit k = 6 for the given rep.

ⓒ
```
  11110001
- 01100011
----------
  10001110
```
(= 128 + 2 + 4 + 8 = 142)

Decimal: $128+64+32+16+1 = 241$

$64+32+2+1 = 99$

$241_{(10)}$

$- \; 99_{(10)}$

$142_{(10)}$

→ Correct result, fit k = 8 for the given rep.

ⓓ +) 110101 ' complement = 001010

Add 1 :
```
  001010
+      1
--------
  001011
```
(Sign-flipping process finish)

⊕ Check: 110101 (2 complement) = -11 ✓

001011 (2 complement) = 11

+)
```
  100011
- 110101
```
→
```
  100011
+ 001011
--------
  101110
```
(= -32 + 8 + 4 + 2 = -18)

Decimal: $100011 = -29_{10}$ (= -32 + 2 + 1)

$001011 = 11$

$-29 + 11 = -18$

→ Correct result, fit k = 6 for the given rep.

ⓔ
```
   100011
+  101101
---------
  1010000
```
The result is correct with 7-bit register (-64 + 16 = -48)

but for 6-bit register, it's incorrect (16)

Decimal: $100011 = -29$

$101101 = -19$

$-29 + (-19) = -48$ (out of range of 6 bit 2's compleme)

ⓓ
```
  00011101
+ 00101010
----------
  01000111
```
(= 64 + 4 + 2 + 1 = 71)

Decimal: $00011101 = 29$ (16 + 8 + 4 + 1)

$00101010 = 42$ (32 + 8 + 2)

$29 + 42 = 71$

→ Correct result, fit k = 8 for given rep.

Question 6

(a)

Pairs	Hamming distance
α, β	5
α, u	3
α, σ	6
β, u	4
β, σ	5
u, σ	7

$\Rightarrow$ Distance 3 code

$\rightarrow$ # of error can be detected : $3-1=2$
# of error can be corrected : $(3-1)/2 = 1$

(b)

Pairs	Hamming Distance
X, Y	6
X, Z	5
Y, Z	4

$\Rightarrow$ Distance 4 code

$\rightarrow$ # of error can be detected : $4-1=3$
or error can be corrected: $(4-1)/2 = 1$ (round down)

(c) +) Each hex digit = 4 binary digits

$\rightarrow$ data bit = $m = 4 \times 2 = 8$ bits (also enough to cover all possible combinations of 2 digit hex)

+) We have $m + r + 1 \leq 2^r$

$\Leftrightarrow 9 + r \leq 2^r \rightarrow r = 4$

+) B6H = $1011\ 0110_2$. see part c1 table

2FH = $0010\ 1111_2$. see part c2 table

(d) +) Each hex digit = 4 binary digits

$\rightarrow$ data bit = $m = 4 \times 4 = 16$ bits

+) We have $m + r + 1 \leq 2^r$

$\Leftrightarrow 17 + r \leq 2^r \rightarrow r = 5$

+) A3F8H = $1010\ 0011\ 1111\ 1000_2$. see part d1 table

C0AEH = $1100\ 0000\ 1010\ 1110_2$. see part d2 table

(e) See the table.

Partc1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
P_1 1		1		0		1		0		1										
	P_2 1	1			1	1			1	1										
			P_4 0	0	1	1					0									
							P_8 0	0	1	1	0									
															P_{16}					

		1		0	1	1		0	1	1	0									
P_1	P_2	m_{15}	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0

3 1's at 3,7,11
$\rightarrow P_1 = 1$

5 1's at 3,6,7,10,11
$\rightarrow P_2 = 1$

2 1's at 6,7
$\rightarrow P_4 = 0$

2 1's at 10,11
$\rightarrow P_8 = 0$

$\rightarrow$ The Hamming code: <u>1</u><u>1</u>1<u>0</u>011<u>0</u>0110
(the underline is for paritybits)

Parte 2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
P_1 0		0		0		0		1		1										
	P_2 1	0			1	0			1	1										
			P_4 0	0	1	0					1									
							P_8 0	1	1	1	1									
															P_{16}					

		0		0	1	0		1	1	1	1									
P_1	P_2	m_{15}	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0

+) For p_1 : two 1's at 9, 10

$\rightarrow p_1 = 0$

+) For p_2 : three 1's at 6, 10, 11

$\rightarrow p_2 = 1$

+) for p_4 : two 1's at 6, 12

$\rightarrow p_3 = 0$

+) For p_8 : four 1's at 9, 10, 11, 12

$\rightarrow p_8 = 0$

$\rightarrow$ The Hamming code : 010001001111

Part d1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
P_1 1		1		0		0		0		1		1		1		1		0		0
	P_2 0	1			1	0			0	1			1	1			1	0		
			P_4 1	0	1	0					1	1	1	1					0	0
							P_8 1	0	0	1	1	1	1	1						
															P_{16} 0	1	1	0	0	0

		1		0	1	0		0	0	1	1	1	1	1		1	1	0	0	0
P_1	P_2	m_{15}	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0

→ The Hamming code : 101 1010 100 11111 011000

Part d2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
P_1 1		1		1		0		0		0		1		1		0		1		0
	P_2 0	1			0	0			0	0			0	1			1	1		
			P_4 0	1	0	0					0	1	0	1					1	0
							P_8 0	0	0	0	0	1	0	1						
															P_{16} 1	0	1	1	1	0

		1		1	0	0		0	0	0	0	1	0	1		0	1	1	1	0
P_1	P_2	m_{15}	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0

→ The Hamming code : 101010000000101101110

Part e1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P_1 0		1		1		0		1		0		0		1		0		0		1		0		1
	P_2 0	1			0	0			1	0			0	1			0	0			1	0		
			P_4 0	1	0	0					0	0	0	1					0	1	1	0		
							P_8 0	1	1	0	0	0	0	1									0	1
															P_{16} 1	0	0	0	0	1	1	0	0	1
0	0	1	0	1	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	1	1	0	0	1
P_1	P_2	m	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0	m	m	m	m

→ All parity bits are correct → no error

+) The databits are : 1100 1100 0010 0001 1001

Using the look up table the hex value is : CC219

Parte 2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
					✓	✓	✓					✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓
7 1's ←	P_1 0		1		0		✓ 0		1		0		1		1		1		0		1		1		0
6 1's ←		P_2 1	1			✓ 1	0			0	0			0	1			1	0			1	1		
				P_4 1	0	1	0					0	1	0	1					1	1	1	1		
3' 1's ←								P_8 0	1	0	0	✓ 0	✓ 1	✓ 0	✓ 1									0	0
																P_{16} 0	1	1	0	1	1	1	1	0	0
	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	1	1	0	1	1	1	1	0	0
	P_1	P_2	m	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0	m	m	m	m

+) Parity bits p_1, p_2, p_8 are incorrect

+) The correct data bit are those checked by p_4 and p_{16}, indicate by the green tick

→ The incorrect bit(s) must be in the list: 3, 4, 9, 10, 11

→ Simplest fix: change bit 11 from 0 → 1

+) The data bit are now: 1010 1010 1011 1011 1100

Using the lookup table, hex value is: AABBC

(f) For part c : length of message is 12 < 16 → condition of no 2 bits closer than 16 bits not satisfied

→ pass .

For part d : See the table in the next page

(see table at the back)

Suppose we want to send message FF00 H = 1111 1111 0000 0000_2

Suppose the error is on bit 3 and 20

→ p_1, p_2 , p_4, p_{16} incorrect (the value of parity bits here is the value when message is correct)

A way to fix : Change bit 19, from 0 → 1

→ p_1 , p_2, p_{16} are correct

Or change bit 21 and bit 2

Change: p_4 , from 0 → 1 (parity bits can be an error too, why not ?)

→ The resulted message : 0111 1111 0000 0110

Conclusion : error can be detected , but may not be correctly fixed

For part e : Since the table is just a minor extension of the table in d

Suppose now the message is FF000 H, the error is the same as described above

The same process happens, where it can detect but not correct error

→ Also cannot work properly

A little bit more on part c : So since the condition was not satisfied, in such a scheme like this, only single-bit error can occur

With that in mind, if we use the table in part c, we see that all data bits are checked by 1-2 parity bits . Since only single-bit error, if a bit is wrong, all the parity bit are wrong , you just need to look at the data bit in common that is checked by the incorrect parity, flip that → corrected

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
							✓	✓	✓	✓	✓	✓	✓	✓						
P_1 1		0		1		1		1		1		0		0		0		1		0
	P_2 1	0			1	1			1	1			0	0			0	1		
			P_4 0	1	1	1					1	0	0	0					1	0
							P_8 0	1	1	1	1	0	0	0						
															P_{16} 0	0	0	1	1	0
		1		1	1	1		1	1	1	1	0	0	0		0	0	0	0	0

P_1	P_2	m_{15}	P_4	m_{14}	m_{13}	m_{12}	P_8	m_{11}	m_{10}	m_9	m_8	m_7	m_6	m_5	P_{16}	m_4	m_3	m_2	m_1	m_0