The Chinese University of Hong Kong

Department of Information Engineering

Summer Workshop – Fun with Information Engineering and Security

Lab 2 – Encoding

Introduction

Encoding is a process that allows us to represent information in a format that can be easily processed, transmitted, or stored by computers and other digital systems. It involves transforming data from its original form into a standardized format that can be understood and interpreted by both humans and machines.

This lab focuses on the concepts of encoding techniques. Participants will explore various encoding methods. Practical exercises are provided to help students understand how these techniques are applied in real-world scenarios.

Section 1: Encoding scheme

Encoding is a process of representing data or information in a specific format. The purpose of encoding is typically to facilitate data storage, transmission, or interpretation.

Section 1.1: Numeral system

In daily life, we use decimal (base10 十進制) to represent a number. Besides decimal, numbers can be represented by different bases, such as binary (base2 二進制) and hexadecimal (base16 十六進制).

In base 2, it represents value in: 0 1

In base 10, it represents value in: 0 1 2 3 4 5 6 7 8 9

In base 16, it represents value in: 0 1 2 3 4 5 6 7 8 9 A B C D E F

Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Base 2	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111
Base 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

e.g. To represent a value 103 of base 10:

In base 10: 123_{10}

In base 2: 1111011_{2}

In base 16: $7B_{16}$

What is the meaning of each digit in base 10?

$$123_{10} = 100 + 20 + 3 = (1 \text{ x } 10^{2}) + (2 \text{ x } 10^{1}) + (3 \text{ x } 10^{0})$$

Can the same expression apply to base 2 and 16 to find the value of base 10?

1111011_2

$= (1 \times 2^6) + (1 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$

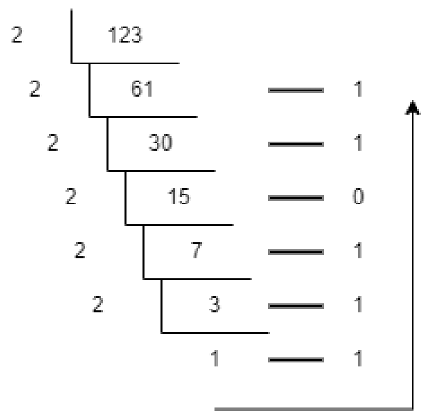
$= 123_{10}$

$7B_{16}$

$= (7 \times 16^1) + (11 \times 16^0)$

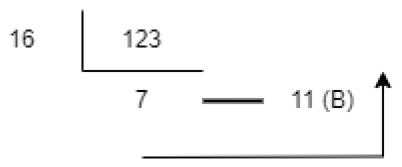
$= 123_{10}$

To convert base 10 to base 2, keep dividing the value by 2 until it is smaller than 2, and mark the remainder of each division:



Reading the value from bottom to top, hence $123_{10} = 1111011_2$

To convert base10 to base 16, keep dividing the value by 16 until it is smaller than 16, mark the remainder of each division:



Reading the value from bottom to top, hence $123_{10} = 7B_{16}$

Try it!

Fill in the blank cell of the table:

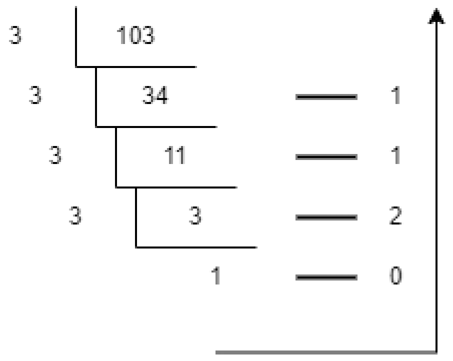
Base 2	Base 10	Base 16
	34_{10}	
		$8E_{16}$
0110001_2		
	362_{10}	

Question:

1. Do larger bases use fewer / more digits to represent the same value?
2. Do larger bases provide advantages/disadvantages in a computer system? Why?

Besides of represneting value using numbers, you can define your own set of alphabet (e.g. A-Z, a-z, symbols !@#$%^). For example, we are representing a value with the set {A, B, C}, where it represents {0, 1, 2} in base 10 respectively:

For example, converting 103_{10} to {A, B, C}. First, compute the base3 value:



We got: 123_{10} = 10211_3

Than, we can map the base3 value to our own set of alphabet:

Base 3	1	0	2	1	1
{A, B, C}	B	A	C	A	A

Hence, the result is: BACAA

To decode the value from {A, B, C} to base 10:

1. Map the value back to base3: BACAA → 102111_3
2. Convert the base3 value back to base10:

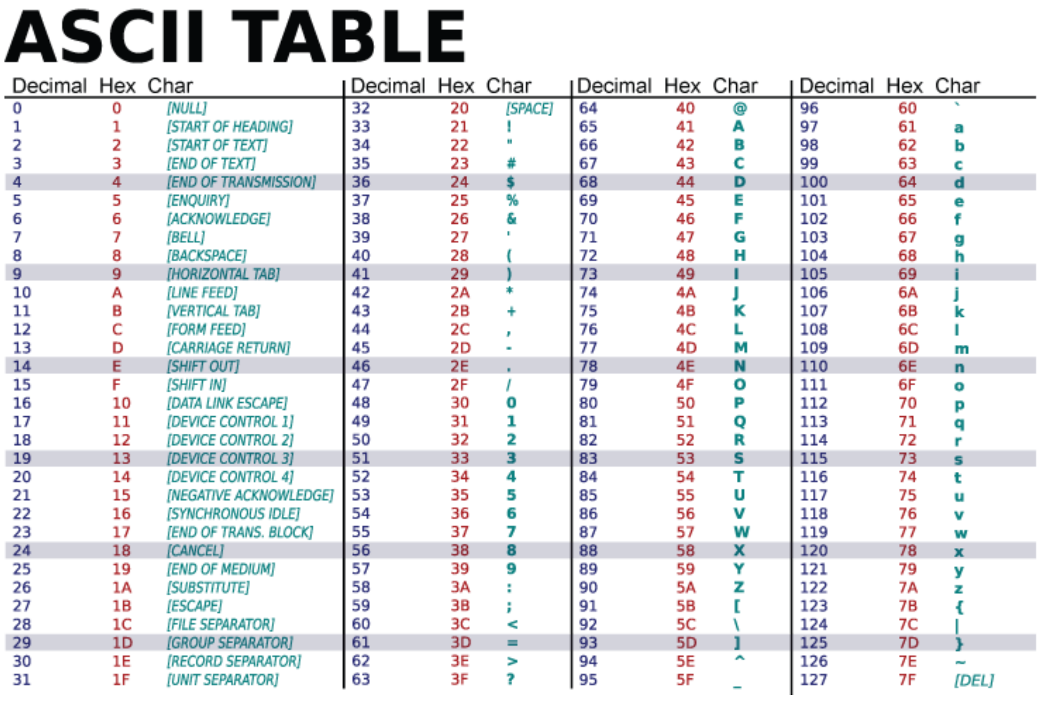
102111_3

$= (1 \times 3^{4}) + (0 \times 3^{3}) + (2 \times 3^{2}) + (1 \times 3^{1}) + (1 \times 3^{0})$

$= 103_{10}$

Section 1.2: Encoding in computer

Besides encoding numbers with different bases, we also encode characters in computer systems. Character encoding schemes are used to represent characters from different writing systems or character sets. They assign unique numeric codes to each character, enabling computers to store, process, and transmit text data. ASCII and Unicode are some of the most well-known character encoding schemes. the following images show the ASCII table:



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Source: https://www.geeksforgeeks.org/ascii-table/

The ASCII encoding scheme defines a set of 128 characters, including control characters (such as newline, carriage return, and tab) and printable characters (including letters, digits, punctuation marks, and special symbols). Each character is assigned a unique numeric value between 0 and 127. Each character can map to a number, e.g. Q can be represented by 81 in decimal.

Try it!

Copy the following code snippet to https://iesummerworkshop.github.io/py-lab.html coding section.

```
print(ord('A'))
print(ord('a'))
```

By using the `ord()` function, the program returns the corresponding Unicode value in decimal value.

1. What is the output of the program?

Copy another code snippet to the coding section.

```
print(chr(65))
```

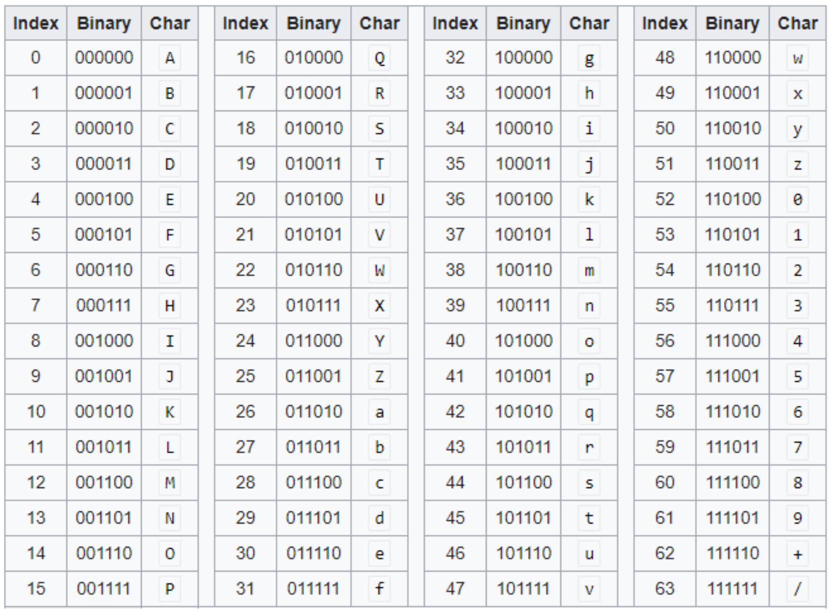
By using the `chr()` function, the program returns the corresponding decimal value in the Unicode table.

1. What is the output of the program?

Base64 is another well-known encoding scheme used in computer systems. It is a binary text encoding scheme that represents binary data in an American Standard Code for Information Interchange (ASCII) string format. It uses a base-64 numeral system. In this system, 64 different characters (digits and symbols) are used to represent the values of the binary data.

Character set of base64: `A-Z`, `a-z`, `0-9`, `+`, `/`

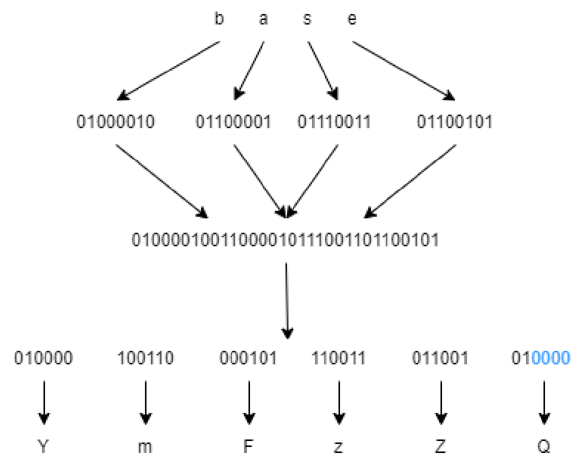
Extra character for padding: `=`



Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	0
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/

Source: https://www.geeksforgeeks.org/encoding-and-decoding-base64-strings-in-python/

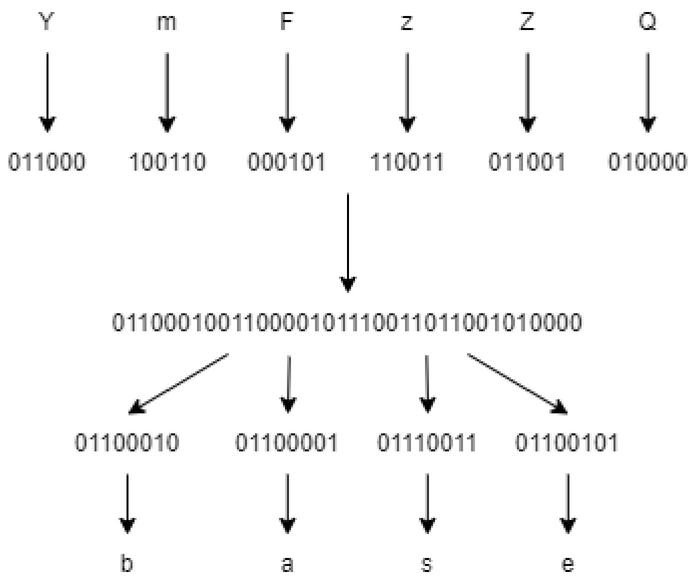
The following are the steps to encode the ASCII string to base64



1. Convert each character into binary representation in ASCII table
2. Divide the binary string into 6-bit chunks
3. Pad the last group if necessary (i.e. if the last chunk is less than 6 bits, add 0 to make it 6 bits)
4. Convert each 6-bit group to the corresponding Base64 character (refer to the Base64 table)
5. Base64 encoded strings must be a multiple of 4 characters. If not, add = at the end

Therefore, the Base64 encoded string for "base" is `YmFzZQ==`

To decode base64 back to ASCII:



1. Remove the padding character =
2. Convert each character into binary representation in the base64 table
3. Divide the binary string into 8-bit chunks
4. Convert each 8-bit group to the corresponding ASCII character (refer to the ASCII table)

Base64 is widely used in computer systems. As some protocols (e.g. email) and data formats (e.g. JSON) are text-based and may not support binary data, Base64 encoding allows binary data (e.g. images, audio) to be embedded in these text-based formats.

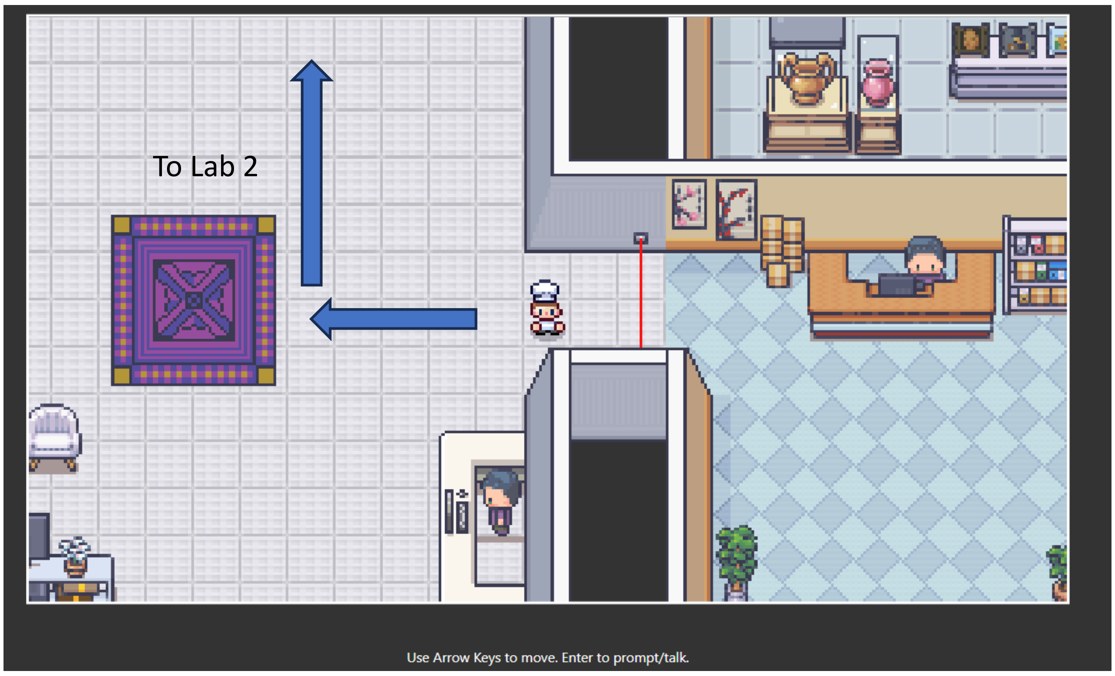
Try it!

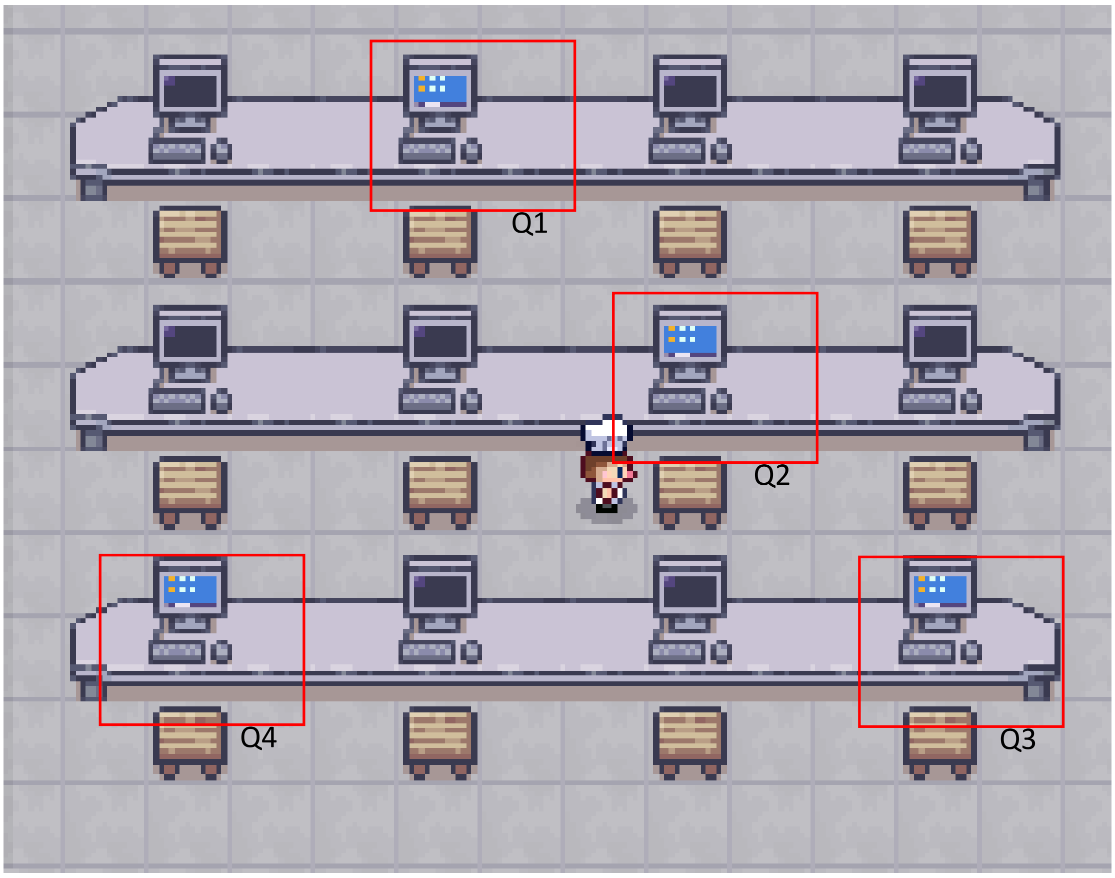
Fill in the blank cell of the table:

ASCII	Base64
Hello	
	SW5mb3JtYXRpb24=

Section 4: Challenge time

Now, please go to https://iesummerworkshop.github.io/rpg/page.html to enjoy your challenge! Walk in front of the computer and press enter to interact with it. Mark down the key you get!





1. **[☆] Base Conversion I**

Complete the function hex_to_dec to convert a decimal number to a hexadecimal number.

2. **[☆ ☆] Base Conversion II**

Complete the function b64_to_ascii to convert text (represented in ASCII) to base64 representation

3. **[☆ ☆ ☆] Base Conversion III**

Complete the function bin_to_dec to convert a binary value to a decimal number. Where it uses 🌕🌑 to represent a value.

4. **[☆ ☆ ☆ ☆ ☆] Base Conversion IV (base39)**

Complete the function base39_to_ascii to convert base39 encoded string to ASCII. Where it uses "0¼½¾123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ" to represent a character set.

Interact with this counter to submit the key for lab 1 and lab 2