


The background is a light gray gradient. It is decorated with numerous water droplets of various sizes, some with soft shadows, giving a sense of depth. In the upper center, there is a faint, circular logo or watermark that appears to contain a stylized plant or sprout.

字元

SPROUT 2018

The background is a light gray gradient. It features several realistic water droplets of various sizes, some with highlights and shadows, scattered across the frame. A faint, large, circular, textured pattern is visible in the upper center, resembling a ripple or a lens flare.

什麼是字元?

ASCII碼

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	:	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

ASCII 碼

- 每一個字母(含數字、特殊符號)，皆有一個特定的對應數字
- 大小寫字母是分開的，視為不同字元

字元	ASCII碼
‘ 0 ’ ~ ‘ 9 ’	48 ~ 57
‘ A ’ ~ ‘ Z ’	65 ~ 90
‘ a ’ ~ ‘ z ’	97 ~ 112

字元 VS 數值

宣告、初始化

- `char c = 'a' ;`
- `char c = 97 ;`

輸入、輸出

- `std :: cin >> c ;`
- `std :: cout << c ;`
- `std :: cout << 'a' ;`

- `int a = 4 ;`

- `std :: cin >> a ;`
- `std :: cout << a ;`
- `std :: cout << 4 ;`

字元 → 數值

- 方法一：

```
int a = ' 0 ' ;
```

```
cout << a ;
```

- 方法二：強制轉換

```
static _ cast < int > ( word );
```

數值 → 字元

- 方法一：

```
char a = 48 ; ( char a = ' 0 ' ; )
```

*浮點數在轉換時會先轉換成整數，再轉成字元

- 方法二：強制轉換

```
static _ cast < char > ( value ) ;
```


既然字元和數值可以互換...

- 是非題：

() 1.字元可以比大小

() 2.字元可以運算

() 3.字元可以拿來寫數學考卷

() 3.字元可以拿來記帳

既然字元和數值可以互換...

- 是非題：

(O) 1.字元可以比大小

(O) 2.字元可以運算

(X) 3.字元可以拿來寫數學考卷

(X) 3.字元可以拿來記帳

當然，如果你認真要拿字元來寫數學考卷，而且不怕被當，或者想不開要拿字元記帳，也不是不行啦~~

字元比大小

- 兩個字元比大小的時候，其實是比他們兩個對應的**ASCII**碼

- E.G. :

‘ a ’ (97) < ‘ b ’ (98)

‘ a ’ (97) > ‘ A ’ (65)

‘ 1 ’ (49) < ‘ 8 ’ (56)

字元運算

- 字元在做運算的時候，其實是拿他們兩個對應的**ASCII**碼當數字運算
- E.G. :

`int a = ' 2 ' + ' 3 ' ;` $(i = 50 + 51 = 101)$

`int b = 2 + ' a ' ;` $(b = 2 + 97 = 99)$

練習

輸入一個字母，輸出它的大/小寫。

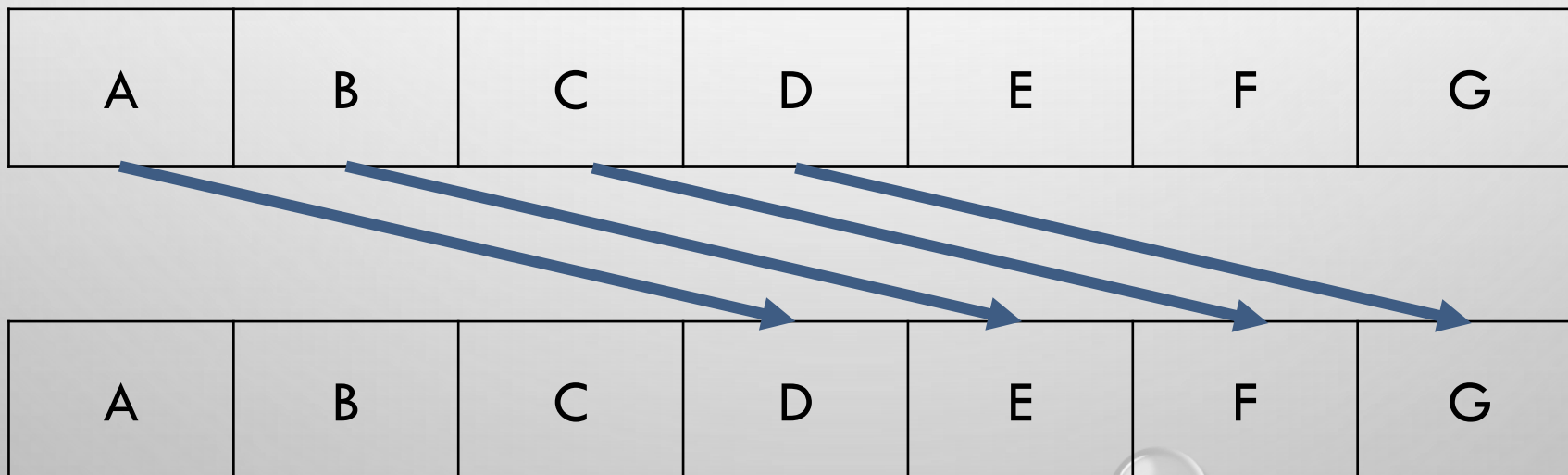
- 範例輸入：a
- 範例輸出：A
- 範例輸入：Z
- 範例輸出：z

凱薩加密

- 凱薩加密是一種替換加密的技術，明文中的所有字母都在字母表上向後（或向前）按照一個固定數目進行偏移後被替換成密文。**(from Wiki)**

凱薩加密

- 凱薩加密是一種替換加密的技術，明文中的所有字母都在字母表上向後（或向前）按照一個固定數目進行偏移後被替換成密文。(from Wiki)
- E.G. 當偏移量為3時，A→D



活動

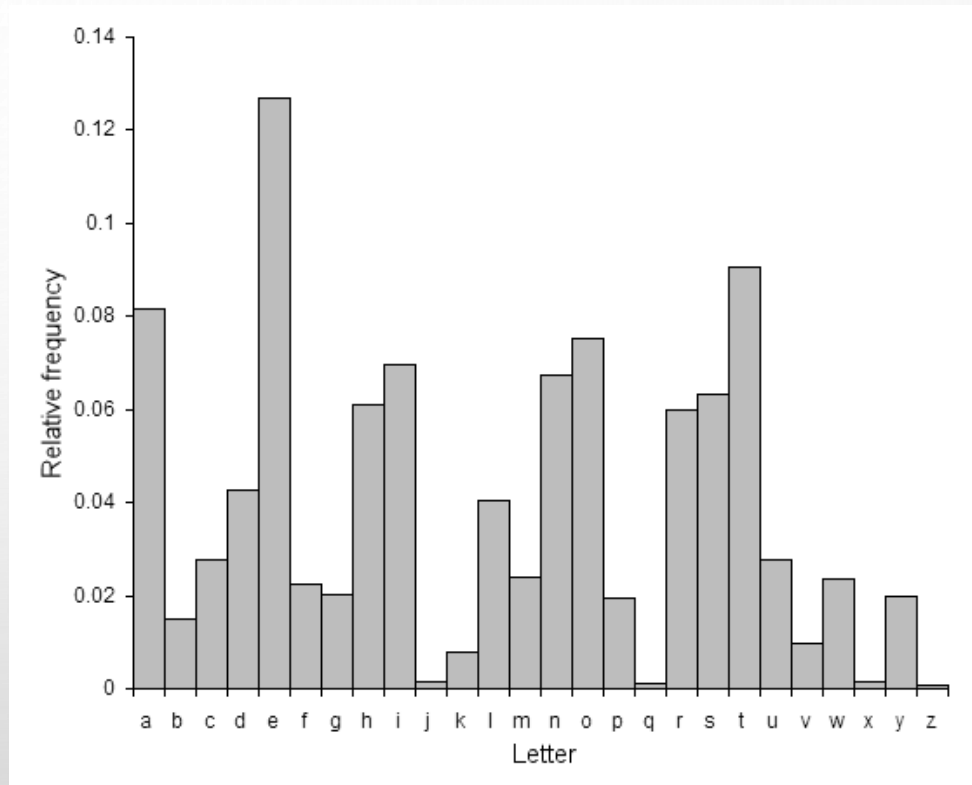
延伸内容

字元函式(使用標頭檔#include<cctype>)

函式	說明
isdigit(ch)	若字元是數字，回傳true
isalpha(ch)	若字元是字母，回傳true
isalnum(ch)	若字元是數字或字母，回傳true
islower(ch)	若字元是小寫字母，回傳true
isupper(ch)	若字元是大寫字母，回傳true
isspace(ch)	若字元是空白，回傳true
tolower(ch)	回傳指定字元的小寫字元
toupper(ch)	回傳指定字元的大寫字元

頻率分析

- 像凱薩加密這樣字母一一對應的加密方法，可以用頻率分析加以破解
- 在英文中，**E**和**T**的使用頻率最高，**Q**和**Z**則偏低
- 分析一段足夠長的密文，得出密文中個字母的出現頻率，再與正常情況對照，即可求解



CC BY-SA 3.0,

<https://commons.wikimedia.org/w/index.php?curid=551867>

改良-維吉尼亞加密

- 加密方式：
- 先選定密鑰，在這裡我們以「**CODE**」為範例，加密；

THE VIGENERE CIPHER IS A FORM OF POLYALPHABETIC SUBSTITUTION.

- 又在凱薩加密中**C**行、**O**行、**D**行、**E**行的替換方式如下表

	A	B	C	D	...	W	X	Y	Z
C	C	D	E	F	...	Y	Z	A	B
O	O	P	Q	R	...	K	L	M	N
D	D	E	F	G	...	Z	A	B	C
E	E	F	G	H	...	A	B	C	D

- 我們輪流使用各行的偏移，得到以下結果
- 密鑰：CODE
- 明文：THE VIGENERE CIPHER IS A FORM OF POLYALPHABETIC SUBSTITUTION.
- 暗文：VVH ZKUHRGGH GKKEKIT WV E HDUQ QT SSNNDPRVDFGILG UJEWVWWYVWRR.
- 一個字元加密後會變成不同的字元，使頻率分析法無效
- 維吉尼亞加密法從16世紀被提出以來，有很長一段時間被認為是無法破解的，直到19世紀才被完全的、有系統的破解

練習

- 請寫一支程式以「SPROUT」為密鑰，加密：

AN ARRAY IS A DATA STRUCTURE CONSISTING OF A COLLECTION OF ELEMENTS.

練習

- 請寫一支程式以「SPROUT」為密鑰，加密：

AN ARRAY IS A DATA STRUCTURE CONSISTING OF A COLLECTION OF ELEMENTS.

- 答案

SD RGMTR XK O XTMP KIMOUJMGY VHDKWNNBDX DZ T UEDZYVMXGC JY
WBVVYHMI

中文字元?

圖例

增益集

連結

註解

文字

符號

?

×

字型(F): (一般文字)

子集合(U): 基本拉丁文

	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_

最近用過的符號(R):

,	o	、	;	:	!	?	「	『	(【	#	%	&	*	※
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Unicode 名稱:

Space

字元代碼(C): 0020

從(M): Unicode (十六進位)

Unicode (十六進位)
ASCII (十進位)
ASCII (十六進位)
繁體中文 Big5 (十六進位)