

CS2315 Computer Programming

Assignment Three (2022-23 Sem. A)

Deadline: **26-Nov-2022 23:59** (Week 13 Sat) No Late submission will be accepted

Problem description

In this assignment, you will work on a trivial encryption program which works by shuffling 70 selected characters in sequence:

<space> character after '0'

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 .,-!?()

The encryption scheme is controlled by 3 integers: **Width**(≤ 70), **Height**(≤ 70) and **Skip**. The basic idea is that the scheme wraps the 70 characters (*in sequence*) above to a rectangle of dimension **W**x**H** in clockwise order starting from the upper-left corner. (You may assume $70 \leq W*H$, no checking is needed). As an example, if **W**=12, **H**=6, **S**=0:

A	B	C	D	E	F	G	H	I	J	K	L
f	g	h	i	j	k	l	m	n	o	p	M
e	4	5	6	7	8	9	0	.	q	N	
d	3)	(?	!	-	,	r	O
c	2	1	z	y	x	w	v	u	t	s	P
b	a	Z	Y	X	W	V	U	T	S	R	Q

Then the characters are extracted *in columns* from left to right, top to bottom order (and skipping all unused blanks):

AfedcbBg432aCh51ZDi6zYEj7)yXfk8(xwGl9?wVhm0!vUIIn -uTJo.,tSKpqrsRLMNOPQ

The encryption works by replacing a character from the original sequence to a character in the new sequence at the same position.

Original	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 .,-!?()
New	AfedcbBg432aCh51ZDi6zYEj7)yXfk8(xwGl9?wVhm0!vUIIn -uTJo.,tSKpqrsRLMNOPQ

For example, 'B' (2nd character in the original sequence) is changed to 'f' (2nd character in the new sequence). Likewise, the message "Hello" will be changed to "g8??H".

The scheme, however has one problem – 'A' (upper-left) is always mapped to 'A'. Therefore the integer **S** is introduced. Instead of wrapping the 70 characters directly,

each character is considered as being preceded by **S** *unused blanks*. (The original scheme with no blank can be considered as **S=0**). As an example, if **W=36**, **H=6**, **S=2**:

	A		B		C		D		E		F		G		H		I		J		K		L
	a		b		c		d		e		f		g		h		i		j		k		l
	y		z		1		2		3		4		5		6		7		8		9		
Z)		(?		!		-		,		.				0		M
	x		w		v		u		t		s		r		q		p		o		n		m
	Y		X		W		V		U		T		S		R		Q		P		O		N

Since **S=2**, character 'A' is preceded by 2 blanks and therefore it's no longer the first character. The new sequence (*ignoring unused blanks*) will be:

ZaxYAybwXBzcvWC1)duVD2(etUE3?fsTF4!grSG5-hqRH6,ipQI7.joPJ8 knOK90lmNLM

In this assignment you can assume $70 \cdot (1+S) \leq W \cdot H$, no checking is needed.

Implementation

You will develop your program based on the template file downloaded from Canvas.

In the file, there is a **Codec** class with the following private members:

int W, H, S	Width, Height, Skip values for the scheme.
char C [71]	The 70 selected characters in sequence.
char M [70][70]	The 2 dimensional matrix used for shuffling.

Your task is to complete the implementation of the following public items:

Codec ()	Default constructor with no shuffling (W=70, H=1, S=0)
Codec (int w, int h, int s)	Parameterized constructor taking W, H, S
void config (int w, int h, int s)	Change the parameters of the encryption system. Could possibly be called multiple times before the program finish.
void showSetting ()	Display the encoding parameters, 2D matrix and the new character sequence in the same format as sample output. The printed matrix should be in dimension of W*H (i.e. no extra trailing spaces before <enter>) and unused blanks in the 2D matrix should be printed as space characters.
void encode (char in[], char out[])	Encode a given cString in [] to another cString out [] using the current codec settings. Characters outside sequence C (e.g. symbols like the '@' mark) will be copied without change.
void decode (char in[], char out[])	Decode a given cString in [] to another cString out [] using the current codec settings. Characters outside sequence C (e.g. symbols like the '@' mark) will be copied without change.

You may add in new variables and functions but you're not allowed to change the existing items in the template (e.g. changing the type or count of function input, or changing public/private settings are not allowed). **<string>** library and **string** class are not allowed. You may use **<iostream>** , **<iomanip>** and **<cstring>** whenever appropriate.

The **main()** function is given in the template. You are not allowed to change the **main()** function. The program will keep reading single-character commands (*in uppercase*) until 'Q' (quit) is encountered.

'C' command is used to configure the encryption system, with integers **W**, **H** and **S** on the same line. Input in PASS is always valid ($70 \cdot (1+S) \leq W \cdot H$).

'S' command is used to display the current setting (which could possibly be changed by the 'C' command). Besides showing **W**, **H**, and **S**, it also displays the 2D matrix and the new character sequence (*of the 70 characters*).

'E' and 'D' commands are used for encryption and decryption respectively. The message is separated from the 'E' / 'D' command by exactly one space character. The message length is less than 1000 character and it may contain spaces or characters outside the selected sequence of 70 characters.

Sample Input / Output:

Example 1: (*User Input is underlined*)

```
Command: S
W=70, H=1, S=0
ABCDEFGHIJKLMN OPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 .,-! ?()
Seq: ABCDEFGHIJKLMN OPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 .,-! ?()

Command: C 12 6 0

Command: S
W=12, H=6, S=0
ABCDEFGHIJKLM
fghijklmnopM
e4567890 .qN
d3 )(?!-,r0
c21zyxwvutsP
baZYXWVUTSRQ
Seq: AfedcbBg432aCh51ZDi6zYEj7)yXFk8(xWG19?wVHm0!vUIn -uTJo.,tSKpqrsRLMNOPQ

Command: E Hello@
g8??H@

Command: Q
Quit
```

Example 2: (User Input is underlined)

```
Command: C 36 6 2
```

```
Command: S
```

```
W=36, H=6, S=2
```

```
  A B C D E F G H I J K L  
a b c d e f g h i j k l  
y z 1 2 3 4 5 6 7 8 9  
Z      ) ( ? ! - , . 0 M  
x w v u t s r q p o n m  
Y X W V U T S R Q P O N
```

```
Seq: ZaxYAybwXBzcvWC1)duVD2(etUE3?fsTF4!grSG5-hqRH6,ipQI7.joPJ8 knOK90lmNLM
```

```
Command: D (sSSKY-5sm
```

```
Well Done!
```

```
Command: Q
```

```
Quit
```

Marking criteria

Submitted program will be tested repeatedly with PASS. Marks will be graded objectively based on the number of correct outputs reported by PASS.

- If the program is not compilable by PASS, zero mark will be given.
- Make sure that the output format (spelling, punctuations, spacing...etc) follows *exactly* the sample output above otherwise PASS will consider your answer incorrect. Note that the marker will make NO manual attempt to check or re-mark program output.

Unlike tutorial exercises, for assignment, PASS will NOT make ALL test cases visible online. (*i.e. there are hidden test cases*). Queries regarding hidden test case(s) will not be entertained. It is the responsibility of a programmer to design own test cases in order to verify the correctness of the written program.

Note that the marking in PASS is automatic, therefore the following situations may lead to extremely low or zero marks:


- Input/output does not match the requirements as defined.
(*e.g. incorrect spacing, incorrect spelling, letter cases or punctuations*)
- Submission of non **.cpp** files (*e.g. .exe or .zip files*)
- Submission with the **“test”** function rather than the **“submit”** function
(*Students may test or submit many times, only the last “submit” will be marked*)

Please also note that the PASS plagiarism check will also be turned on. **The same disciplinary action will be applied without distinguishing which one is the source / copier.** Please safeguard your files if you work on your assignment using public computers (e.g. CityU Lab)

Testing and Submission

Students should submit cpp file to PASS before the deadline. **No other submission method (e.g. hardcopy, email...etc) is accepted.** Students may use whatever IDE/compiler for development, but programming using non-standard, platform-specific features (which is not compilable in PASS/MS Visual Studio) will lead to zero mark. The marker will make no attempt to fix the syntax error and make the code compilable.

If you observe that the answer / runtime in PASS is different from the one you get in your PC, **most likely it is caused by programming bugs like uninitialized variables.** The result by PASS will be used for marking without consideration of what you observe in your local PC.

It is advised that students test the programs with PASS (*using the  Test function*) before final submission (*with the “submit” button*). Be warned that the system could be extremely busy and may become sluggish in responding near deadline. Only **.cpp** files are accepted. Do not submit **.obj** , **.exe** or any other files.

To protect yourself, it is advised that you write down your particulars (full name, student number, eid...) in the beginning of your source code *as comment*.