Memory stack for first_memtest.cpp

start

simpleCharManager simplest_mem_manager; //constructor, now **add #1** to stack //**gets removed after function completion (out of scope)**

char* location = simplest mem_manager.alloc_chars(13);
//would call the function, **add #2** to stack,
//**gets removed after function completion (out of scope)**

*(location+0) = ‘H’   //these would assign to buffer[x] accordingly
*(location+1) = ‘e’
*(location+2) = ‘l’
*(location+3) = ‘l’
*(location+4) = ‘o’
*(location+5) = ‘ ’
*(location+6) = ‘w’
*(location+7) = ‘o’
*(location+8) = ‘r’
*(location+9) = ‘l’
*(location+10) = ‘d’
*(location+11) = ‘!’
*(location+12) = ‘\n’

simplest_mem_manager.free_chars(location+6);
//continue on other stack, now **add #3 to stack**
//**gets removed after function completion (out of scope)**

char* location2 = simplest_mem_manager.alloc_chars(11);
//calls function, now **add #4 to stack**
//**gets removed after function completion (out of scope)**

*(location2+1) = ‘m’
*(location2+2) = ‘o’

*(location2+3) = ‘o’
*(location2+3) = ‘n’
*(location2+4) = ‘!’
*(location2+5) = ‘ ’
*(location2+6) = ‘B’
*(location2+7) = ‘y’
*(location2+8) = ‘e’
*(location2+9) = ‘.’
*(location2+10) = ‘\n’

end

1. Initial Empty Buffer

int BUF_SIZE = 10000;
char buffer[10000];
char* free_place;
buffer[0] = ‘\0’;   //at constructor
buffer[1] = ‘\0’;
...
buffer[9999] = ‘\0’;
free_place = buffer;

2. Allocate space for “Hello World!\n” — it would return pointer pointing to buffer space 0 since it is empty, then insert “Hello World!\n” starting at pointer to buffer[0] in this case and, ending at space 12 into the buffer

free_place = buffer (/* new free place if allocation succeeds */);

3. To remove “world!\n”, use free_chars(pointer_to_buffer[6]) to free/set to ‘\0’ the space starting at buffer[6] to the end of the buffer

buffer[6] = ‘\0’
buffer[7] = ‘\0’;

```
...
buffer[9999] = '\0';
```

4. Allocate space for “moon! Bye.\n” — it would return pointer pointing to buffer space 6 since it is the first location that can allocate 11 spaces (inclusive of first location) for the 11 characters in “moon! Bye.\n”, then insert “moon! Bye.\n” starting at space 6, ending at space 16

```
free_place = &buffer[6] (/* new free place if allocation succeeds */);
```

Buffer at #1

\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	...	\0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	...	[10000]

Buffer after assigning things in main stack after #2

H	e	l	l	o		W	o	r	l	d	!	\n	\0	\0	\0	\0	\0	...	\0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	...	[10

Buffer at free_chars() at #3

H	e	l	l	o		\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	\0	...	\0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	...	[10000]

Buffer after assigning things in main stack after #4

H	e	l	l	o		m	o	o	n	!		B	y	e	.	\n	\0	...	\0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	...	[10000]