1		1
Name:	Login:	IN-CLASS EXERCISE

Address syntax 2

SOLUTION

```
#include <stdlib.h>
int main(int argc, char* argv[]) {
          ash[] = "bin";
    char
                   "den":
    char* cal
    int
          eel
                          // ASCII value for 'f'
    int
                 = 0x66;
          fad
    int*
          gas
                 = &fad;
    int** ȟem
                 = &gas;
    return EXIT_SUCCESS;
}
```

- The type of ash[0] is char
- 2. The type of &cal[0] is char*
- 3. The number base of eel is irrelevant/moot
- 4. The type of *gas is int
- 5. The type of &gas is int**
- 6. The type of **hem is int

Complete each sentence above.

For #1, #2, #4, #5, #6, the answer should be a type, like this:

O. The type of eel is int . // EXAMPLE

This is not a quiz.

These questions originally comprised Quiz 1 in Spring 2019.

This semester, we are using them as an in-class exercise.

Supplement to Solution

The analysis below was *not* a necessary part of success on this in-class exercise. The questions asked only for the <u>type</u> of various expressions. Determining the type does not require knowing the value of an expression or the contents of memory. This table is provided only to aid those who were using the example code from this exercise as a basis for understanding the contents of memory.

Stack						
addr	type*	name*	value	part	fn	
200	int	argc	1	araum anta	_	
204	char**	argv	→ {"./foo"}	arguments	nai	
212	void*			return addr.	main()	
220	char[4]	ash	{98, 105, 110, 0} ⇔ {'b', 'i', 'n', '\0'} ⇔ "bin"	local vars		
224	char*	cal	600			
232	int	eel	101			
236	int	fad	102			
240	int*	gas	236			
248	int**	hem	240			
256						

Неар				
addr	value	r a n		
400				

Data segment			
addr value			
600	{100, 101, 110, 0} ⇔ {'d', 'e', 'n', '\0'} ⇔ "den"		
604			

^{*} Note: type and name are not actually stored in memory. They are listed here only for our understanding. Assume sizeof(int) == 4, sizeof(char) == 1, and sizeof(void*) == 8. The program was run as "./foo".