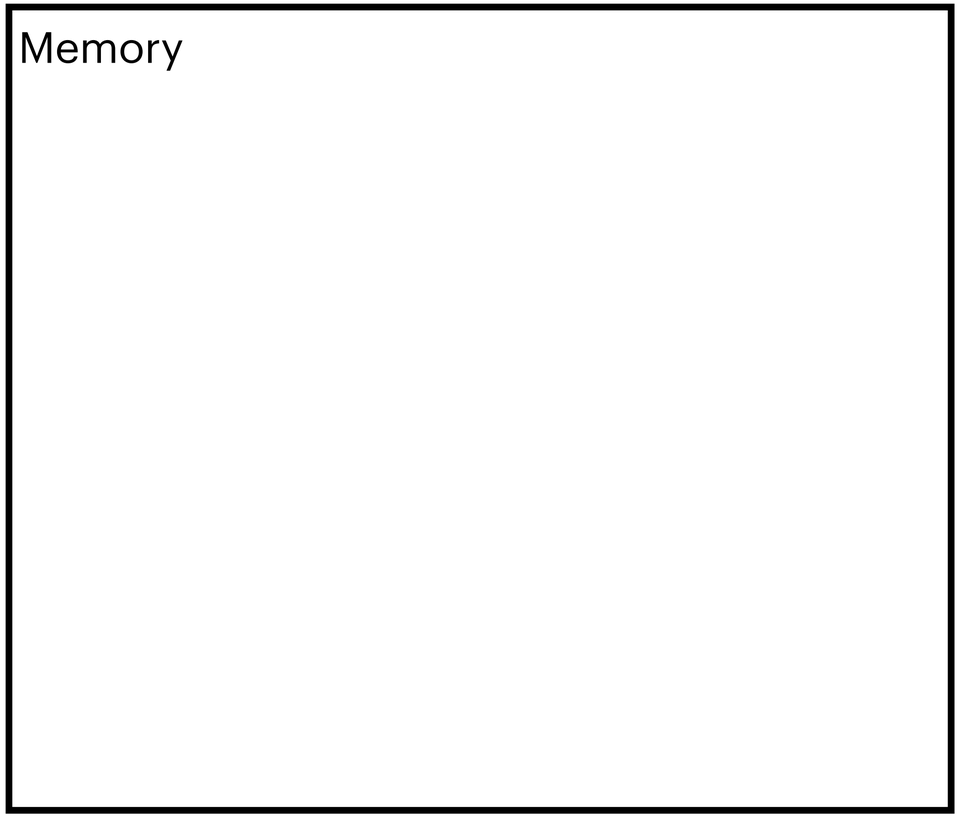
Pointer Syntax

What's happening in this code snippet?

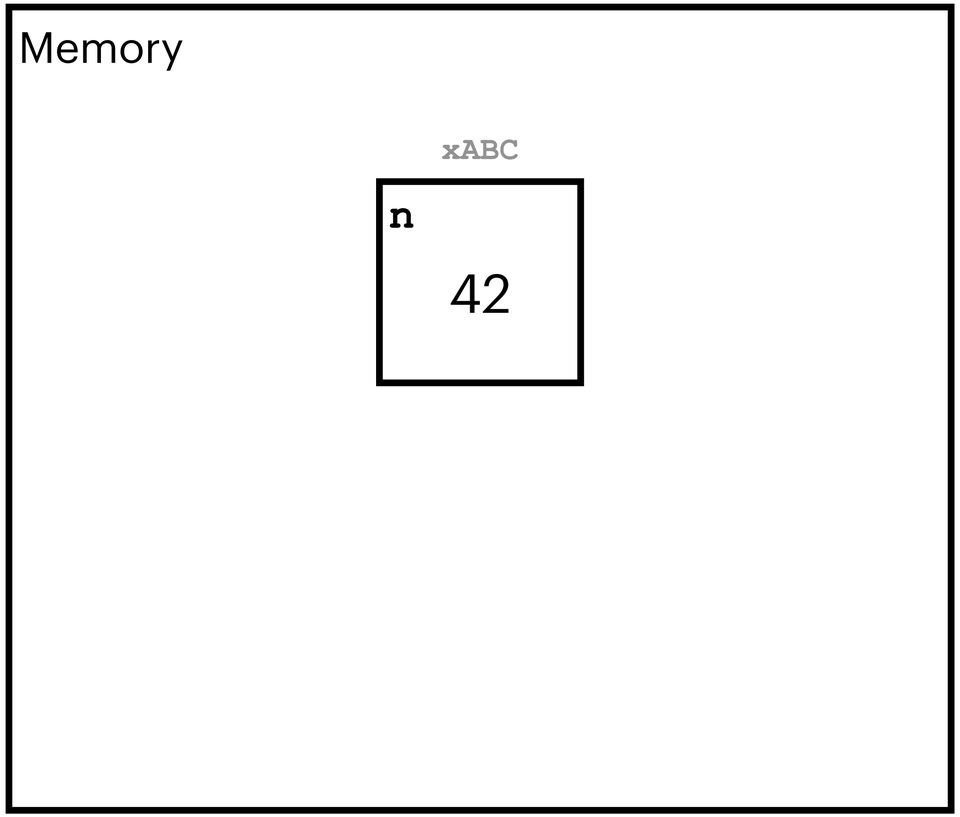
```
int n = 42;
int *p;
int *q;
p = &n;
*p = 5;
*q = 17;
q = p;
*q = 8;
```



Pointer Syntax

What's happening in this code snippet?

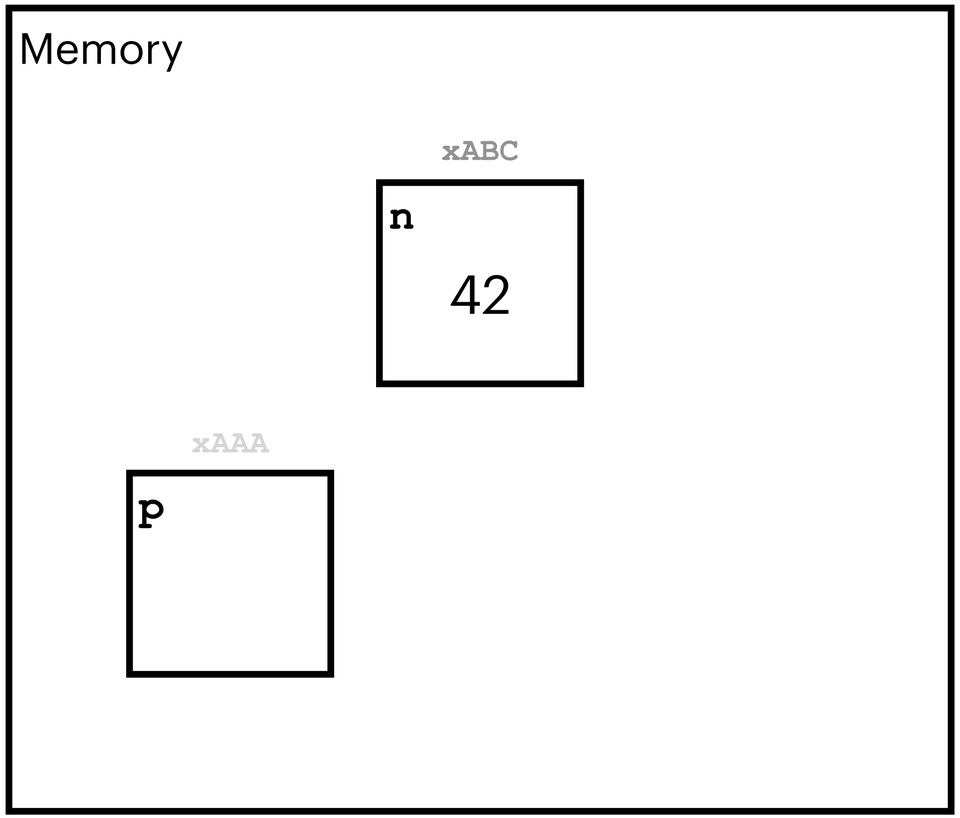
✅
```
int n = 42;
int *p;
int *q;
p = &n;
*p = 5;
*q = 17;
q = p;
*q = 8;
```



Pointer Syntax

What's happening in this code snippet?

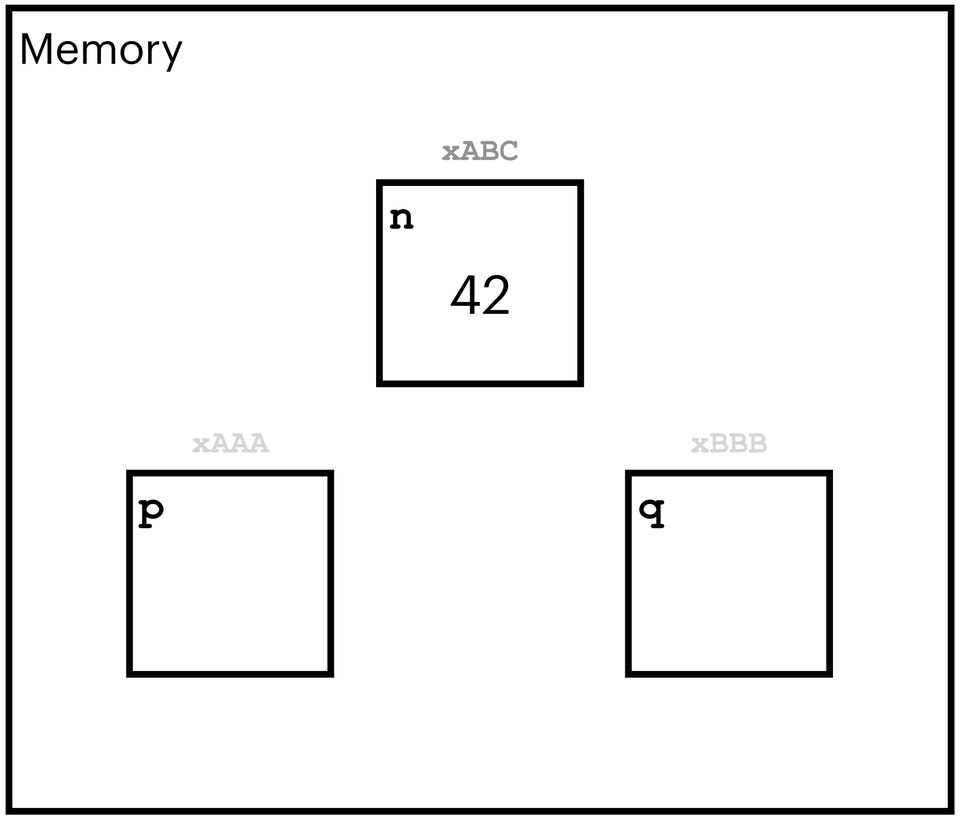
```
✅ int n = 42;
✅ int *p;
   int *q;
   p = &n;
   *p = 5;
   *q = 17;
   q = p;
   *q = 8;
```



Pointer Syntax

What's happening in this code snippet?

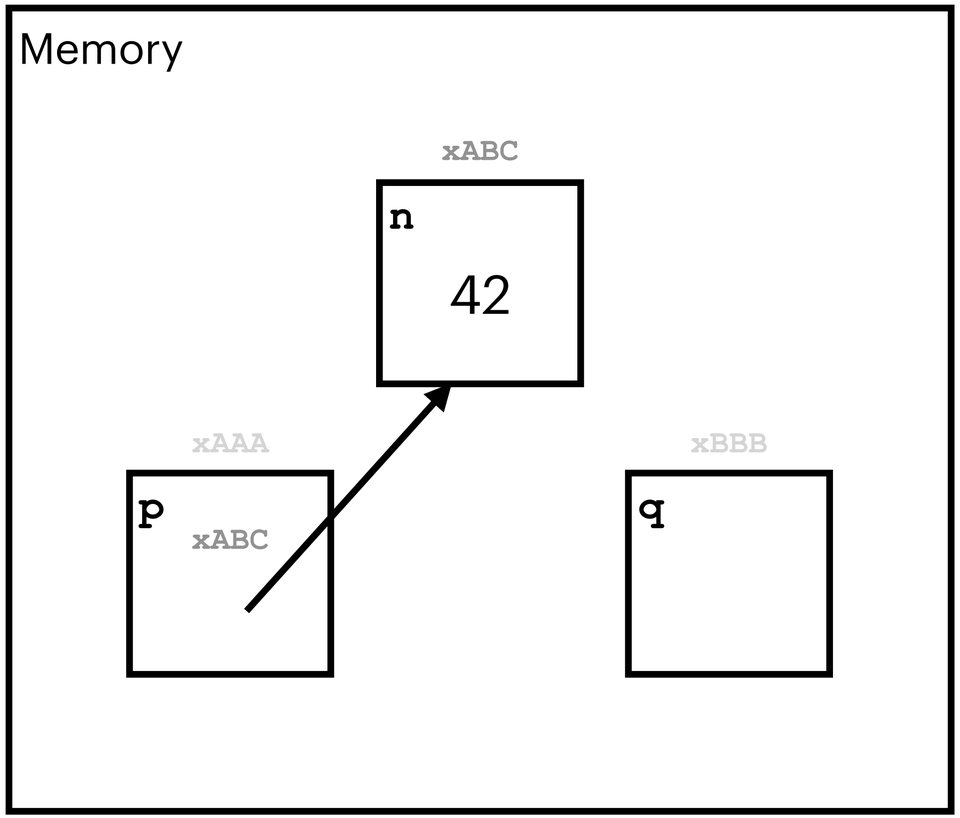
```
✅ int n = 42;
✅ int *p;
✅ int *q;
   p = &n;
   *p = 5;
   *q = 17;
   q = p;
   *q = 8;
```



Pointer Syntax

What's happening in this code snippet?

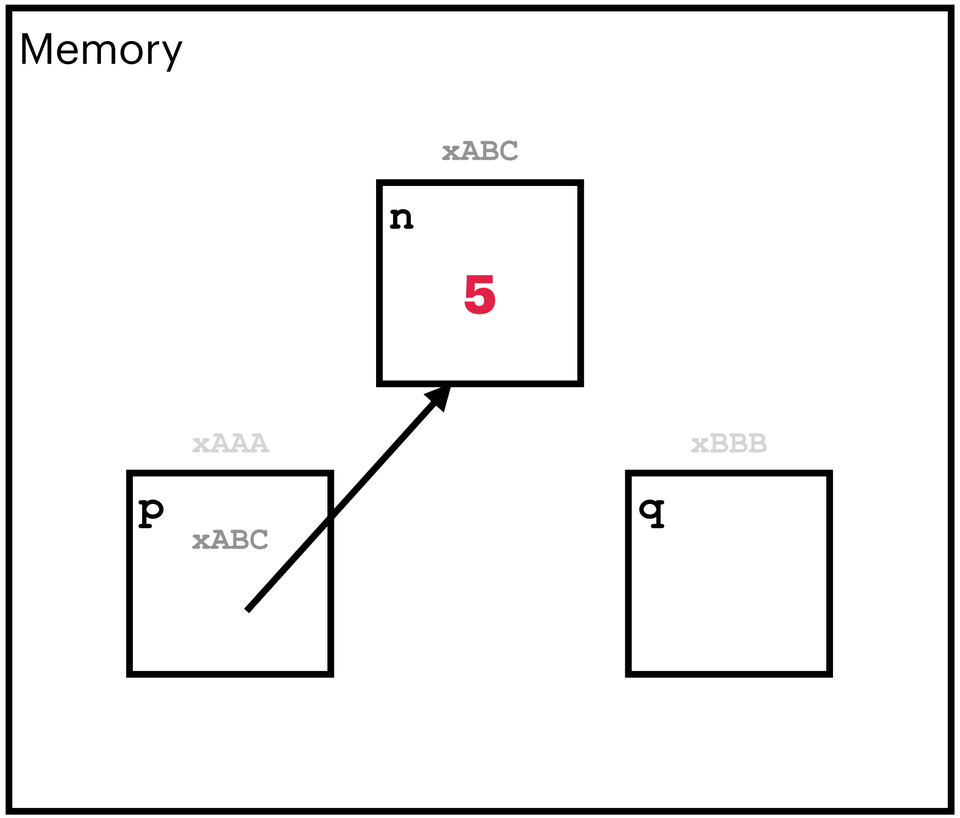
```
✅ int n = 42;
✅ int *p;
✅ int *q;
✅ p = &n;
   *p = 5;
   *q = 17;
   q = p;
   *q = 8;
```



Pointer Syntax

What's happening in this code snippet?

```
✅ int n = 42;
✅ int *p;
✅ int *q;
✅ p = &n;
✅ *p = 5;
   *q = 17;
   q = p;
   *q = 8;
```

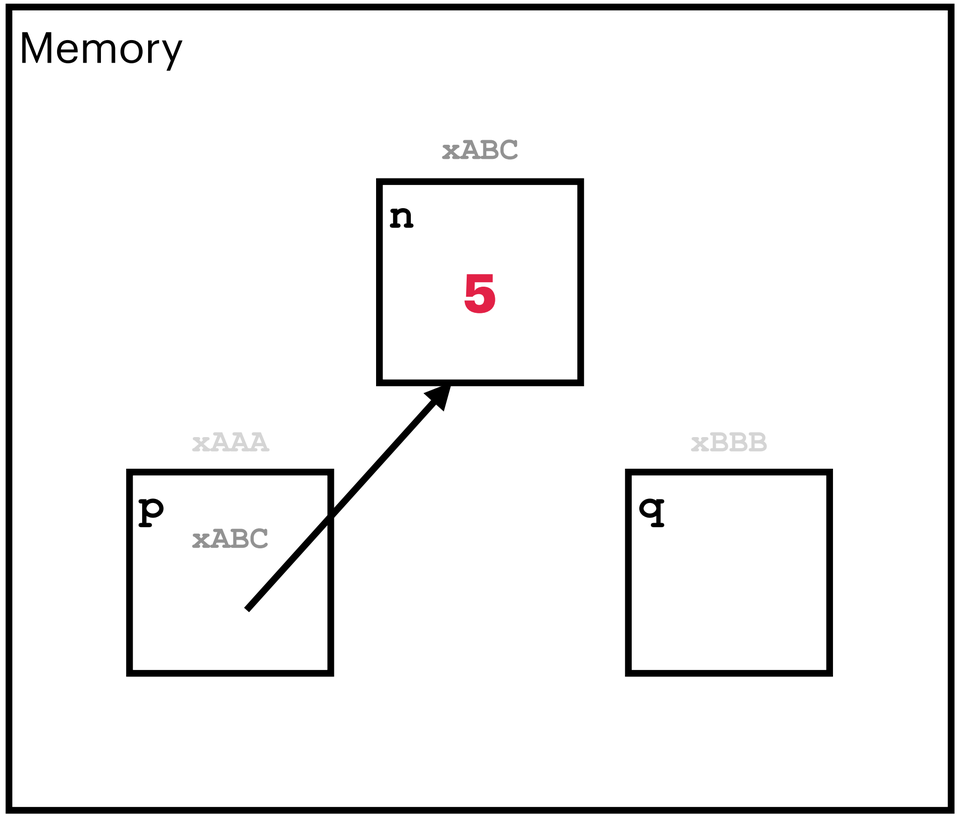


Pointer Syntax

What's happening in this code snippet?

```
✅ int n = 42;
✅ int *p;
✅ int *q;
✅ p = &n;
✅ *p = 5;
❌ *q = 17;
   q = p;
   *q = 8;
```

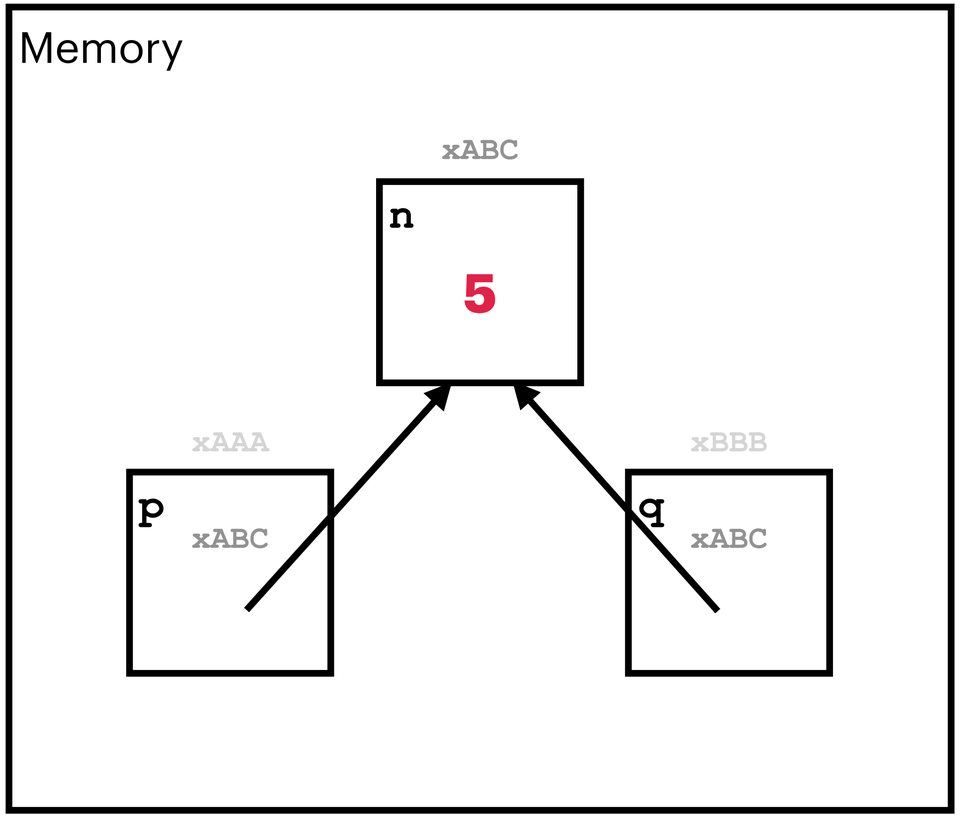
(q doesn't store an address yet)



Pointer Syntax

What's happening in this code snippet?

✅ `int n = 42;`
✅ `int *p;`
✅ `int *q;`
✅ `p = &n;`
✅ `*p = 5;`
❌ `*q = 17;` (q doesn't store an address yet)
✅ `q = p;`
`*q = 8;`

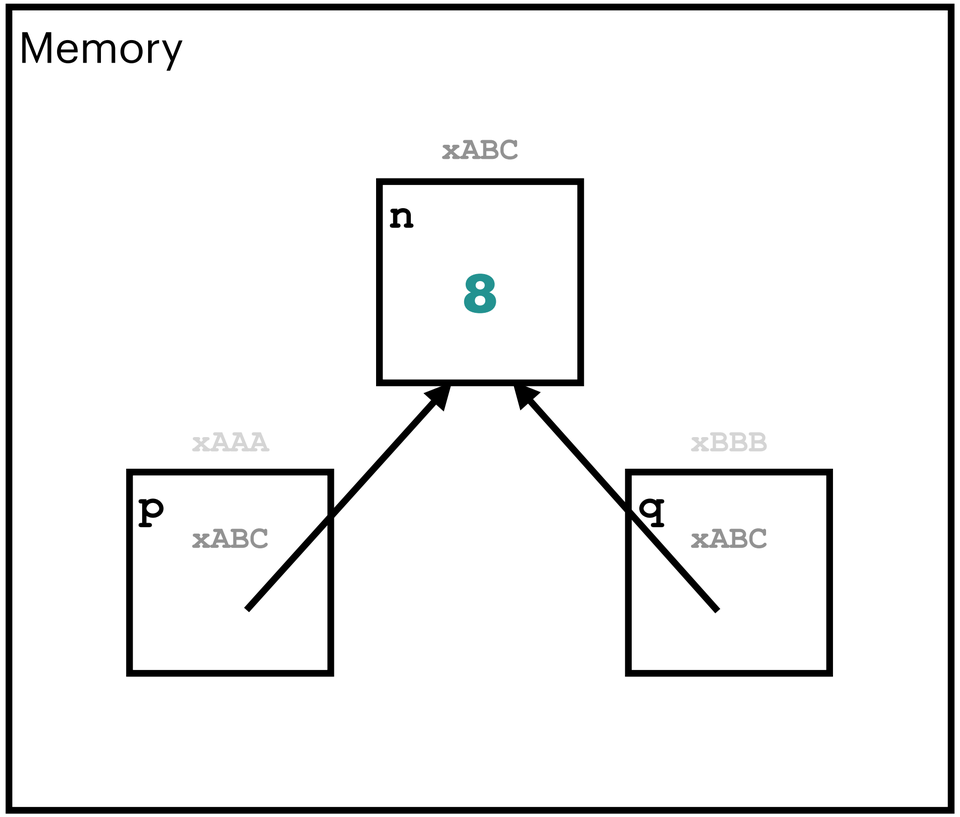


Pointer Syntax

What's happening in this code snippet?

```
✅ int n = 42;
✅ int *p;
✅ int *q;
✅ p = &n;
✅ *p = 5;
❌ *q = 17;
✅ q = p;
✅ *q = 8;
```

(q doesn't store an address yet)



fgets

inspecting the man pages

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <stdio.h>

```
char *
fgets(char * [redacted] str, int size, FILE *[redacted] stream);

char *
gets(char *str);
```

DESCRIPTION

The **fgets**() function reads at most one less than the number of characters specified by size from the given stream and stores them in the string str. Reading stops when a newline character is found, at end-of-file or error. The newline, if any, is retained. If any characters are read and there is no error, a `\0' character is appended to end the string.

The **gets**() function is equivalent to **fgets**() with an infinite size and a stream of stdin, except that the newline character (if any) is not stored in the string. It is the caller's responsibility to ensure that the input line, if any, is sufficiently short to fit in the string.

RETURN VALUES

Upon successful completion, **fgets**() and **gets**() return a pointer to the string. If end-of-file occurs before any characters are read, they return NULL and the buffer contents remain unchanged. If an error occurs, they return NULL and the buffer contents are indeterminate. The **fgets**() and **gets**() functions do not distinguish between end-of-file and error, and callers must use feof(3) and ferror(3) to determine which occurred.