

What is the size of `p` in this code?

```
int *p;
```

8 bytes

4 bytes

16 bytes

The size of the pointer `p` depends on the architecture of the system it's being run on. Specifically, it depends on the size of the address space of the system:

- 1 On a 32-bit system, pointers typically occupy 4 bytes.
- 2 On a 64-bit system, pointers typically occupy 8 bytes.

The code itself doesn't specify the architecture, so we can't definitively say the size of `p` without that context. Both "4 bytes" and "8 bytes" could be correct, depending on the system. "16 bytes" is not typical for pointer sizes on commonly used architectures as of my last update in January 2022.

What is the size of `*p` in this code?

```
int *p;
```

8 bytes

4 bytes

16 bytes

The variable `p` is a pointer to an integer. The size of `*p` (the integer value being pointed to by `p`) will depend on the size of an `int` on the system the code is running on.

For most systems:

- 1 On a 32-bit system, `int` typically occupies 4 bytes.
- 2 On a 64-bit system, `int` usually still occupies 4 bytes, but this can vary based on the system and compiler settings.

Given the options provided:

The most common size for an `int` is "4 bytes". However, it's important to note that the exact size can vary depending on the specific system and compiler. But "16 bytes" is very uncommon for the size of an `int`, and "8 bytes" is possible on some systems but less typical.

What is the size of `p` in this code?

```
int **p;
```

8 bytes

4 bytes

16 bytes

The variable `p` in the code is a pointer to a pointer to an integer (`int **p;`). The size of the pointer `p` depends on the architecture of the system it's being run on, specifically on the size of the address space of the system:

- 1 On a 32-bit system, pointers typically occupy 4 bytes.
- 2 On a 64-bit system, pointers typically occupy 8 bytes.

Regardless of what the pointer points to (whether it's a plain `int` or another pointer), the size of the pointer itself depends on the system's architecture.

Given the options:

- On a 32-bit system, the size of `p` would be "4 bytes".
- On a 64-bit system, the size of `p` would be "8 bytes".

"16 bytes" is not typical for pointer sizes on commonly used architectures as of my last update in January 2022.

What is the size of `*p` in this code?

```
int **p;
```

16 bytes

4 bytes

8 bytes

The variable `p` is a pointer to a pointer to an integer (`int **p`). When you dereference `p` with `*p`, you get a pointer to an integer (`int *`).

The size of this pointer (`int *`) will depend on the system architecture:

- 1 On a 32-bit system, pointers typically occupy 4 bytes.
- 2 On a 64-bit system, pointers typically occupy 8 bytes.

So, given the options:

- On a 32-bit system, the size of `*p` would be "4 bytes".
- On a 64-bit system, the size of `*p` would be "8 bytes".

"16 bytes" is not typical for pointer sizes on commonly used architectures as of my last update in January 2022.

From ChatGPT, designed with  FancyGPT