Memory

Key-Terms

Bit

Short for **binary digit**, a bit is a fundamental unit of information in Computer Science that represents a state with one of two values, typically **0** and **1**.

Any data stored in a computer is, at the most basic level, represented in bits.

Byte

A group of eight **bits**. For example, **01101000** is a byte.

A single byte can represent up to **256** data values (2^8).

Since a **binary number** is a number expressed with only two symbols, like **0** and **1**, a byte can effectively represent all of the numbers between 0 and 255, inclusive, in binary format.

The following bytes represent the numbers 1, 2, 3, and 4 in binary format.

```
1: 00000001
2: 00000010
3: 00000011
4: 00000100
```

Fixed-Width Integer

An integer represented by a fixed amount of **bits**. For example, a **32-bit integer** is an integer represented by 32 bits (4 bytes), and a **64-bit integer** is an integer represented by 64 bits (8 bytes).

The following is the 32-bit representation of the number 1, with clearly separated bytes:

```
00000000 00000000 00000000 00000001
```

The following is the 64-bit representation of the number 10, with clearly separated bytes:

```
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00001010
```

Regardless of how large an integer is, its fixed-width-integer representation is, by definition, made up of a constant number of bits.

It follows that, regardless of how large an integer is, an operation performed on its fixed-width-integer representation consists of a constant number of bit manipulations, since the integer is made up of a fixed number of bits.

Memory

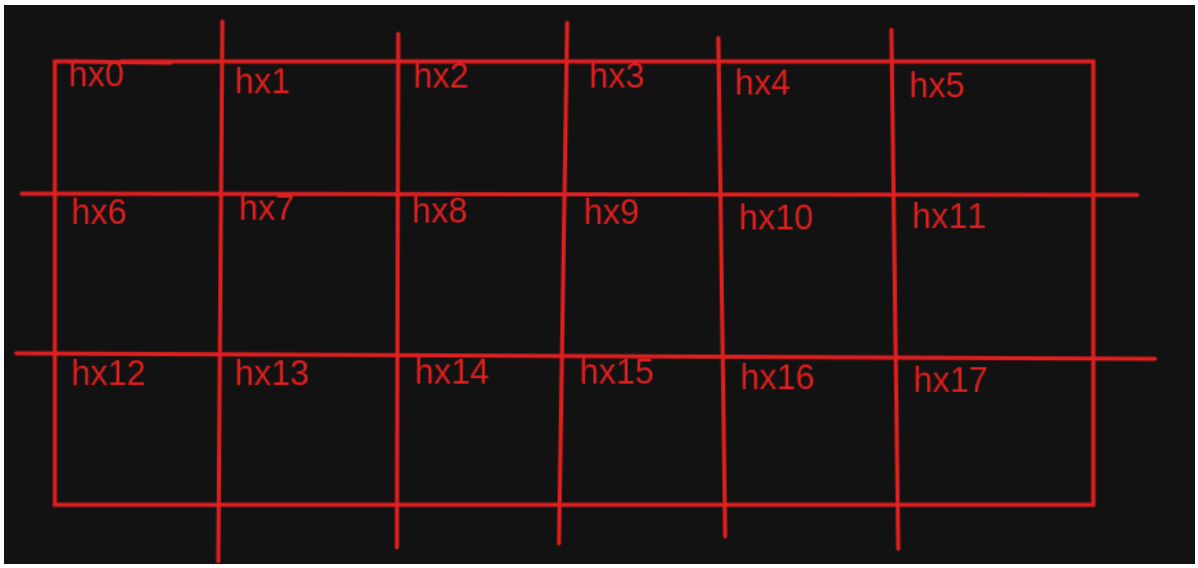
Broadly speaking, memory is the foundational layer of computing, where all data is stored.

In the context of coding interviews, it's important to note the following points:

- Data stored in memory is stored in bytes and, by extension, bits.
- Bytes in memory can "point" to other bytes in memory, so as to store references to other data.
- The amount of memory that a machine has is bounded, making it valuable to limit how much memory an algorithm takes up.
- Accessing a byte or a fixed number of bytes (like 4 bytes or 8 bytes in the case of **32-bit** and **64-bit integers**) is an elementary operation, which can be loosely treated as a single unit of operational work.

Geral

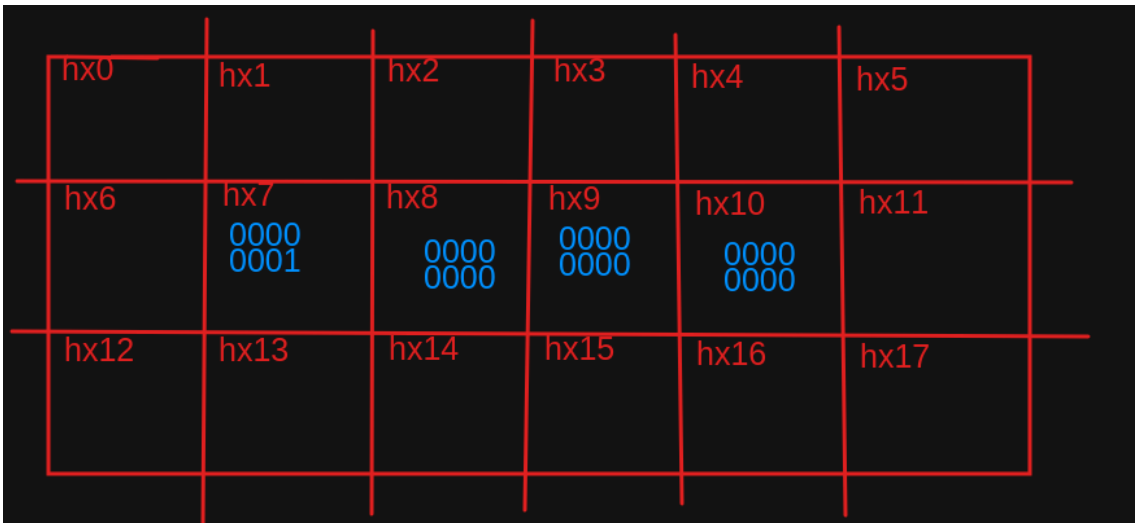
- a memória do computador é composta por slots de memória



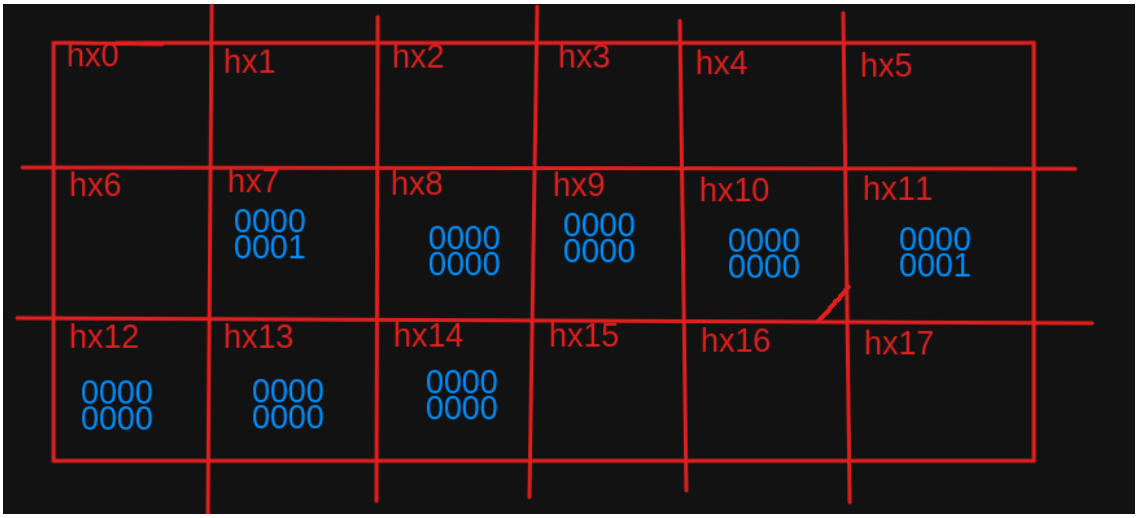
- cada slot comporta 8 bit
  - um bit é um 0 ou 1 (binário)
  - uma sequência de 8 bits é um byte
    - ex: 0010 0011 (é um byte)

Alocação

- em uma arquitetura na qual um int possui 32 bits, é necessário 4 bytes contínuos para alocar um inteiro
  - salvar o valor 1, ficaria:

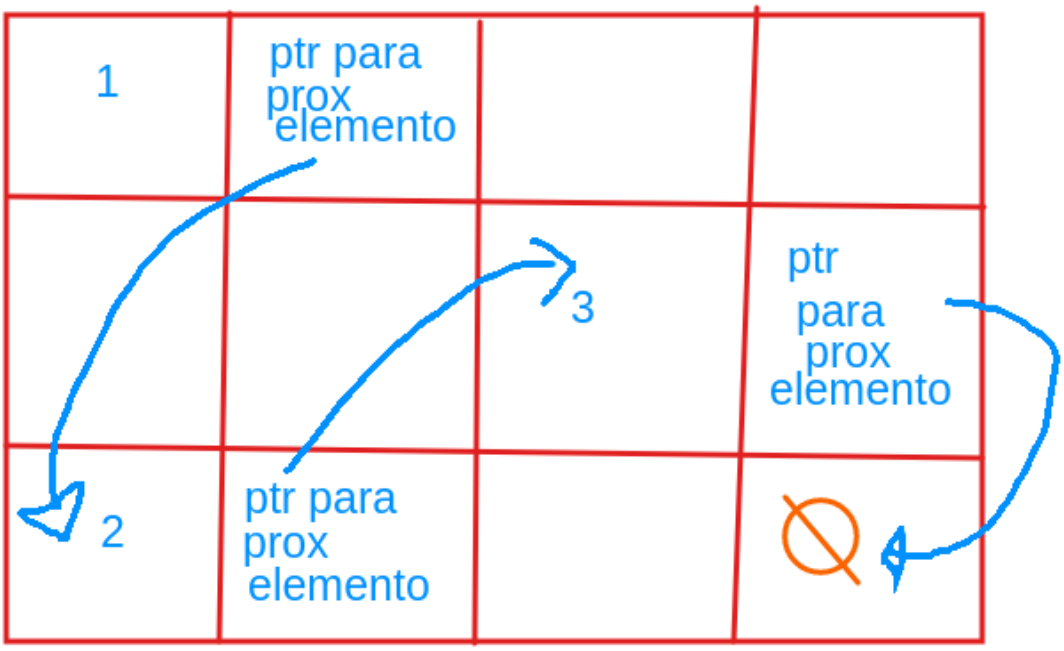


    - não seria possível alocar o valor se não houvesse 4 slots de memória contínuos livres
      - o valor só será colocado em memórias vazia
  - salvar um array de inteiros [1, 1]



    - os valores do array devem estar de forma contínua na memória

- salvar uma lista [1, 2 3]



- os valores de uma lista não precisam estar de forma contínua na memória pois se utiliza ponteiros (slot de memória que guarda o endereço para outro slot)
  - mas é necessário que o nó, ou seja, valor + ponteiro, estejam em memória contínua

```
nó_lista{
  int valor;
  int *próximo;
}
```

cada nó que compõe essa lista consegue guardar o valor e o endereço do próximo elemento da lista