DADOS DE ENTRADA: VETOR DE 40 BYTES.

DADOS DE SAÍDA: VETOR DE 40 BYTES ORDENADO.

TIPOS DE DADOS: VALORES EM BYTES.

ESTRUTURA DE HARDWARE: SUPONHAMOS QUE SEJA UM COMPUTADOR ATUAL.

ESTRUTURA DE SOFTWARE: NO FIM, SUPONHAMOS QUE SEJA ESCRITO EM C.

```
MAIN ( ARGUMENTOS ... ) {
  // PONTEIRO P/ O VETOR DE BYTES DO TIPO PONTEIRO P/ INTEIROS DE 1 BYTE
  VAR* VETOR;
  VETOR = 0X25 // FAZ O PONTEIRO APONTAR PARA DETERMINADO LUGAR NA MEMÓRIA PRÉ-EXISTENTE.

  // O LAÇO A SEGUIR PERCORRE O VETOR DO PRIMEIRO ENDEREÇO ATÉ O PENÚLTIMO, POIS O
  // ÚLTIMO VAI JÁ ESTA ORDENADO AO FINAL
  PARA ( i = 0 ATÉ (40 - 1); i++) {
    // A CADA ITERAÇÃO A VARIÁVEL MIN RECEBERÁ O ÍNDICE CORRESPONDENTE AO ENDE-
    // REÇO DE MEMÓRIA QUE SERÁ USADO PARA COMPOS OUTROS ELEMENTOS.
    MIN = i;

    // SEGUNDO LAÇO PERCORRE O VETOR, SÓ QUE SEMPRE DE UM ÍNDECE A FRENTE
    // DO LAÇO SUPERIOR, PARA COMPARAR OS VALORES SALVOS NOS ENDEREÇO.
    PARA (J = (i+1) ATÉ 40; J++) {
      // A ESTRUTURA DE DECISÃO VERIFICA SE ALGUM VALOR É MENOR QUE O
      // INDICADO NO ENDEREÇO DO ÍNDICE MIN, CASO SEJA, O ÍNDICE É TROCADO.
      SE (*(VETOR+J) < *(VETOR + MIN)) {
        MIN = J;
      } // FIM SE
    } // FIM PARA

    // APÓS O SEGUNDO ~~FOR~~ LAÇO, A ESTRUTURA DE DECISÃO VERIFICA SE É PRECISO
    // TROCAR OS VALORES DE LUGAR, CASO O VALO NO ENDEREÇO INDICADO POR
    // MIN SEJA DIFERENTE, ENTÃO EXISTE UM VALOR MENOR QUE O INDICADO
    // POR I
    SE (*(VETOR+i) != *(VETOR + MIN)) {
      temp = *(VETOR + i)
      *(VETOR + i) = *(VETOR + MIN) // ESSA NOMECLATURA EM TODO
      *(VETOR + MIN) = temp          // ALGORITMO INDICA EM QUE
    } // ~~END~~ FIM SE                // POSIÇÃO NO CONJUTO ESTÁ.
  } // FIM LAÇO PARA                  // EX: ~~*(vetor+0~~ VETOR+0 = 0X25
} // FIM DA FUNÇÃO                    // EX: VETOR+1 = 0X26...
```