Exemplo de arrays dinâmicos usando realocação de memória

Uma versão mais elaborada do que foi dado nas aulas:

```
#include <stdio.h>
#include <stdlib.h>
typedef struct arrayd {
   void **mem;
   int num;
   int tam_max;
   int incr;
   int tam_elem;
} ARRAY;

/**
   Esta funcao cria a estrutura de controle do array dinamico
   @param tam_elem o tamanho de cada elemento
   @param incr quanto o array vai crescer de cada vez
   @return a estrutura de controle e NULL se algo correr mal na alocacao de memoria
*/
ARRAY *criar(int tam_elem, int incr) {
   ARRAY *a;
   a = (ARRAY *) malloc(sizeof(ARRAY)); /* alocar espaco para a estrutura, se correr mal o malloc
devolve NULL */
   if(a != NULL) { /* Se consegui alocar entao inicializar os valores e alocar a area de trabalho inicial
*/
      a->tam_elem = tam_elem;
      a->tam_max = a->incr = incr;
      a->mem = (void *) malloc(a->tam_elem * a->tam_max);
      if(a->mem == NULL) { /* Se nao consegui alocar a area de trabalho liberto o que aloquei e vou
devolver NULL */
         free(a);
         a = NULL;
      }
   }
   return a;
}

/**
   Esta funcao insere um elemento na cauda do array e se for necessario aloca mais memoria
   A memoria alocada e sempre em funcao do numero de elementos, de cada vez que seja
necessario o array cresce para poder armazenar mais incr elementos
   Cuidado: Esta funcao esta a guardar o apontador para o array e nao a copiar a sua proria copia.
Tente modificar a funcao para contemplar esse novo caso
   @param a A estrutura do array
   @param elem O apontador para o elemento
   @return o indice do elemento ou -1 se houver erro
*/
```

```
int insere_elem(ARRAY *a, void *elem) {
    int erro = 0;
    int idx = -1;

    if(a->tam_max == a->num) { /* e preciso alocar mais espaco porque atingimos o tamanho maximo
*/
        int tam = a->tam_max + a->incr;
        void *novo = (void *) realloc(a->mem, tam * a->tam_elem);
        if(novo != NULL) {
            a->mem = novo;
            a->tam_max = tam;
        } else
            erro = 1;
    }

    if(!erro) {
        idx = a->num;
        a->mem[a->num] = elem;
        a->num++;
    }

    return idx;
}

int main() {
    ARRAY *a = criar(sizeof(int), 20); /* criar o array dinamico em que cada elemento ocupa o
tamanho de um inteiro em que o array incrementa o seu tamanho de 20 elementos de cada vez */
    int i;
    int idx = 0;

    for(i = 1000; idx != -1 && i > 0; i--) {
        int *p = (int *) malloc(sizeof(int));
        *p = i;
        idx = insere_elem(a, p); /* porque e que em vez destas 3 linhas nao poderiamos simplesmente
ter idx = insere_elem(a, &i); tente perceber porque */
    }

    for(i = 0; i < a->num; i++) {
        int *p = (int *)a->mem[i];
        printf("%03d\t%d\n", i, *p);
    }

    return 0;
}
```