

## In realla esistano due gunzioni agginative mella libreria:

Voce 025

- void \*calloc(size\_t nmemb, size\_t size): restituisce
  un'area di memoria tale da contenere un vettore di nmemb elementi,
  ciascuno di dimensione size. La memoria è inizializzata a zero. Viene
  verificato se nmemb \* size provoca overflow, caso in cui viene
  generato un errore.
- CALLOC

  è una variante della

  malloc che ni permette

  oli ellocare vettori oli oggetti.

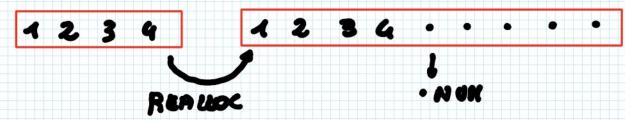
  CALLOC RESTITULISE um Vettore

  di oggetti AZZERATO!

  LA NALOC NO.
- void \*realloc(void \*ptr, size\_t size): viene "modificata" la quantità di memoria puntata da ptr, effettuando una nuova allocazione e spostando size byte da ptr al nuovo buffer allocato. Il buffer puntato da ptr viene liberato con free(ptr). Il nuovo buffer viene restituito dalla funzione.

REALLOC

## NUOVA RIALLOCAZIONE.



chiama maller, nencopy, wa gree, e restituise il viono indiricos;

4 del mono vertape;

#### MAIN. C

# VECTOR.C

VECTOR H

```
### A Process of the Community of the Co
```

```
return 0;
```

### VECTOR. H

```
#pragma once
#include <stdlib.h>
#include <stdbool.h>

struct array;

extern void *init_vector(size_t nmemb, size_t size);
extern bool insert_at(struct array *arr, size_t pos, void *el);
extern bool push_back(struct array *arr, void *el);
extern bool read_from(struct array *arr, size_t pos, void *buffer);
extern bool extract_from(struct array *arr, size_t pos, void *buffer);
extern void free_vector(struct array *arr);
```

```
memory(from, st, arr-esizet);

arranguative;

assortiarescount or arr-esizet);

(starts from);

bool read_frow(front array varr, size_t pes, void wboffer)

("Spec earr-occount) {

memory(boffer, (obr share-parr + pes * arr-esize, arr-esizet);

/ "Start field;

return false;

bool extract_from(front array varr, size_t pes, void wboffer)

(day affec, sti)

size_t size;

size_t size;

size_t size;

(from of (obr share-parr + pes, void wboffer)

(from of (obr share-parr);

field;

field;

field;

field;

arr-esizet;

field;

arr-esizet;

field;

arr-esizet;

field;

arr-esizet;

field;

arr-esizet;

field;

arr-esizet;

arr-esizet;

field;

arr-esizet;

arr-esizet
```

```
CFLAGS=-Wall -Wextra -g
TARGET=dyec
SRCS=main.c vector.c
OBJS=$(SRCS:.c=.o)

%.o:%.c

$(CC) $(CFLAGS) $< -c -o $@

all: $(OBJS)
$(CC) $(CFLAGS) $(OBJS) -o $(TARGET)

.PHONY:clean
clean:
-$(RM) $(TARGET) $(OBJS)
```