Esercitazione di Reti degli elaboratori

Prof.ssa Chiara Petrioli

SAPIENZA
UNIVERSITÀ DI ROMA

Corso di C

Christian Cardia, Gabriele Saturni

Esercitazione

Esercizio 1

Scrivere un programma con le seguenti funzioni:

int *crea_array(int dimensione);

che crea un array dinamico di dimensione "dimensione" e ritorna un puntatore al primo elemento.

void riempi_array(int *array, int dim);

che consente all'utente di riempire l'array

void stampa_array(int *array, int dim);

che stampa a video i numeri salvati nell'array.

Il programma deve (tramite l'utilizzo delle tre funzioni): creare l'array, permettere all'utente di memorizzarci dei numeri interi e infine stamparli.

Esercizio 1 Soluzione (1/2)

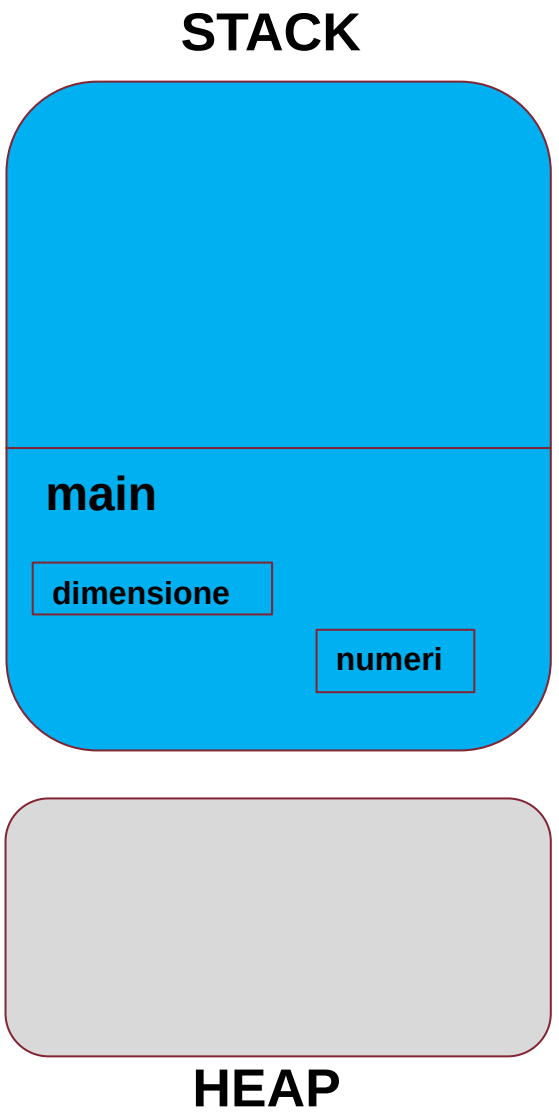
```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```

Esercizio 1 Soluzione (2/2)

```
24  int *crea_array(int dimensione){
25      int *array = NULL;
26      if (dimensione <= 0)
27          return array;
28      array = malloc(dimensione * sizeof(int));
29      return array;
30  }
31
32  void riempi_array(int *array, int dim){
33      for(int i=0; i<dim; i++){
34          printf("\nInserisci elemento %d: ",i+1);
35          scanf("%d",&array[i]);
36      }
37  }
38
39  void stampa_array(int *array, int dim){
40      printf("\nNumeri inseriti:");
41      for(int i=0;i<dim;i++)
42          printf("\n%d",array[i]);
43  }
```

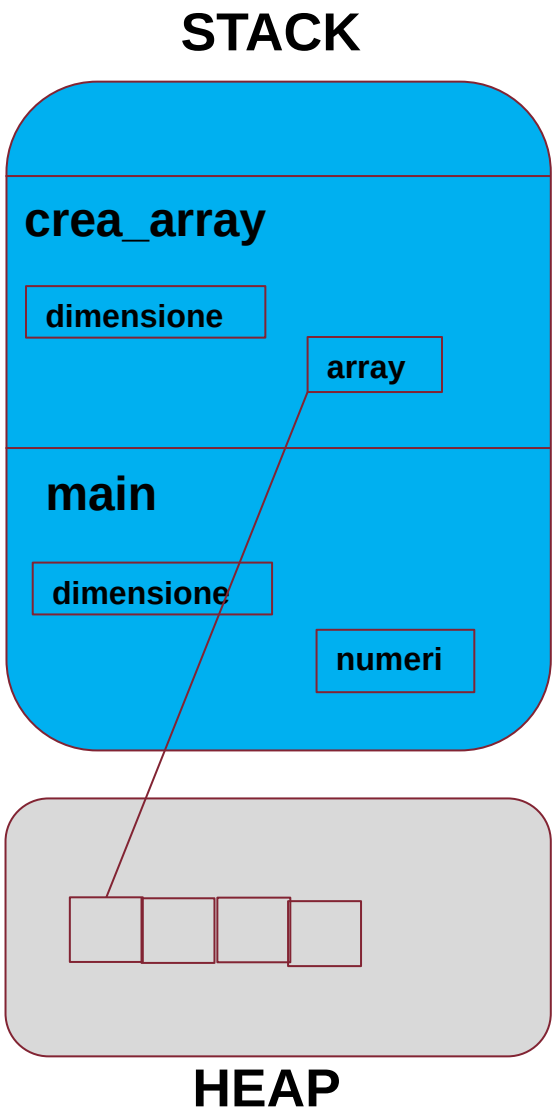
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



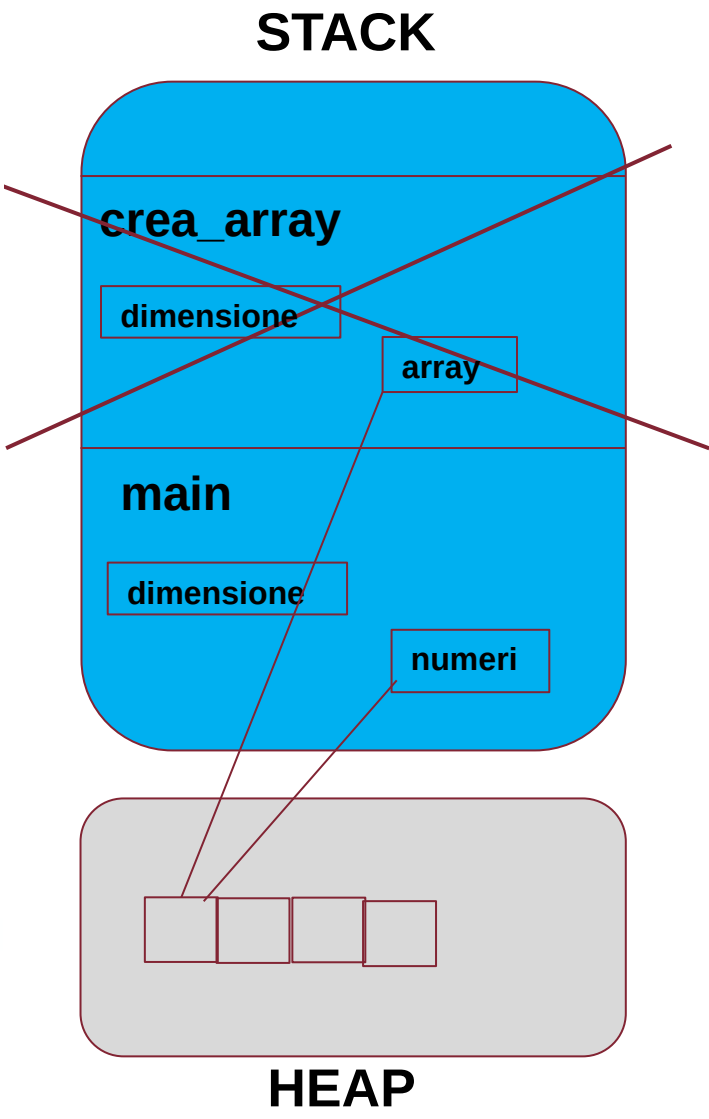
Esercizio 1 Cosa succede nella memoria...

```
24  int *crea_array(int dimensione){
25      int *array = NULL;
26      if (dimensione <= 0)
27          return array;
28      array = malloc(dimensione * sizeof(int));
29      return array;
30  }
```



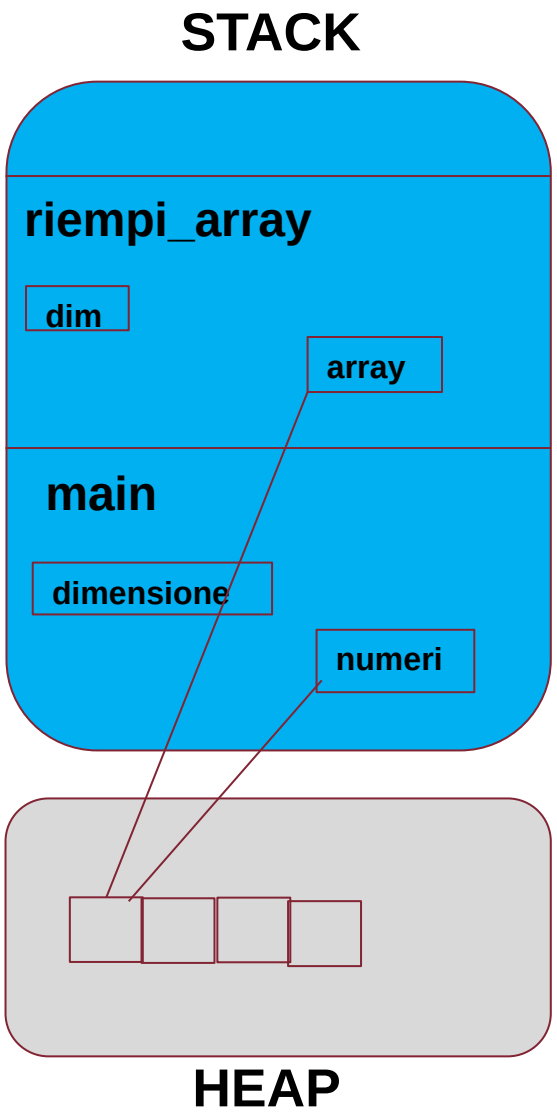
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



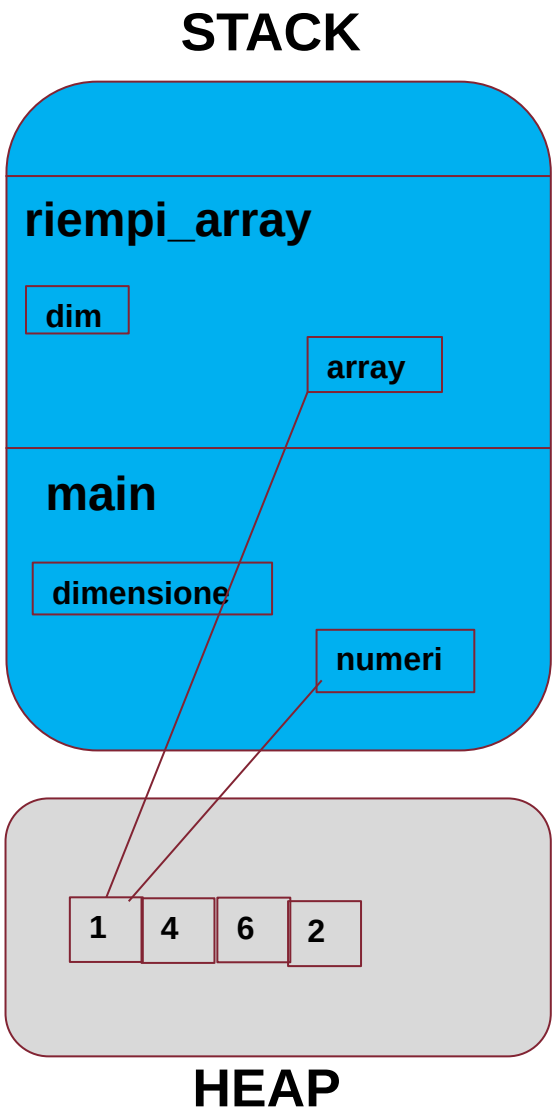
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



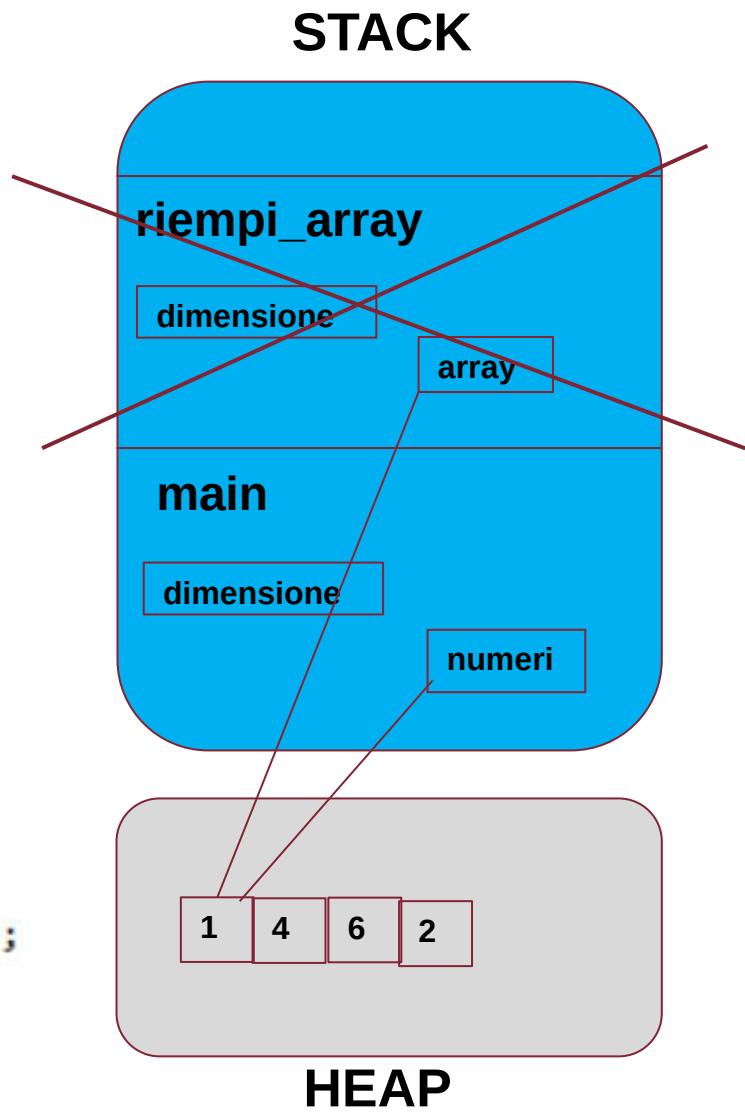
Esercizio 1 Cosa succede nella memoria...

```
32  void riempi_array(int *array, int dim){
33      for(int i=0; i<dim; i++){
34          printf("\nInserisci elemento %d: ",i+1);
35          scanf("%d",&array[i]);
36      }
37  }
```



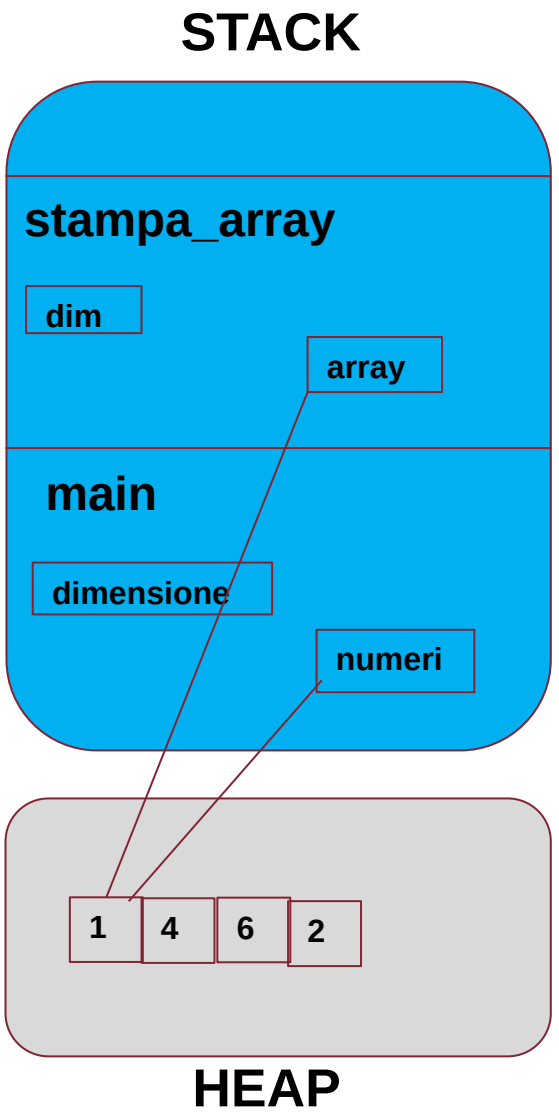
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



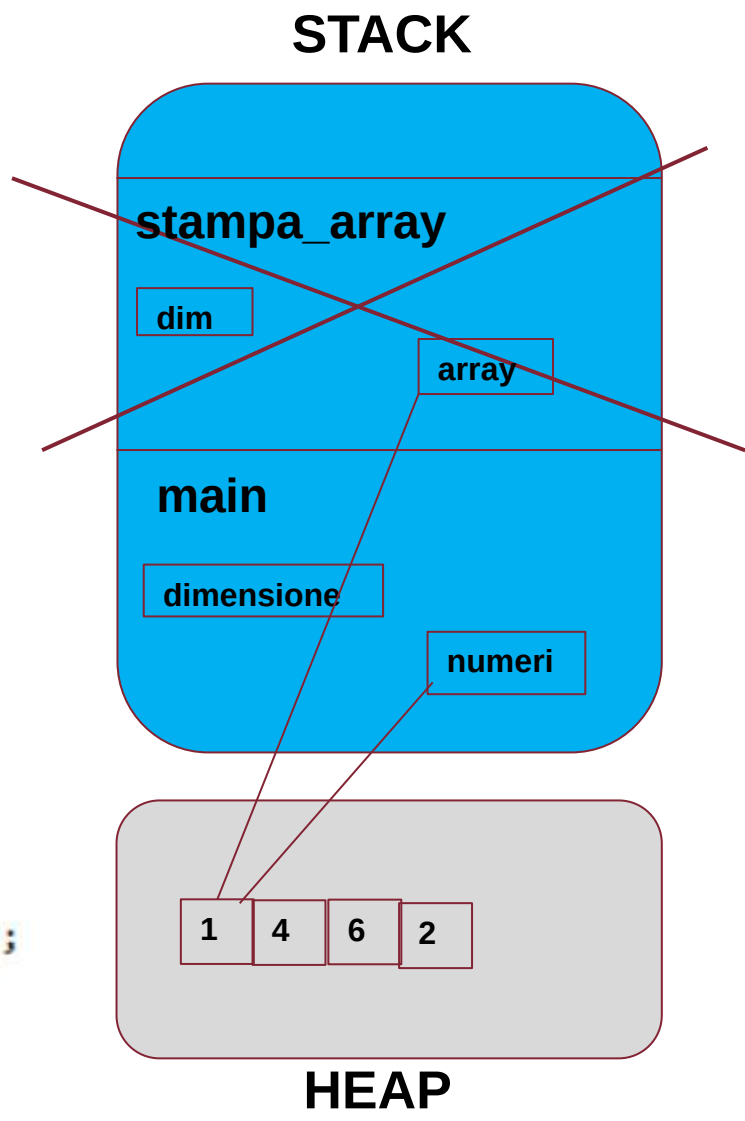
Esercizio 1 Cosa succede nella memoria...

```
39  void stampa_array(int *array, int dim){
40      printf("\nNumeri inseriti:");
41      for(int i=0;i<dim;i++)
42          printf("\n%d",array[i]);
43  }
```



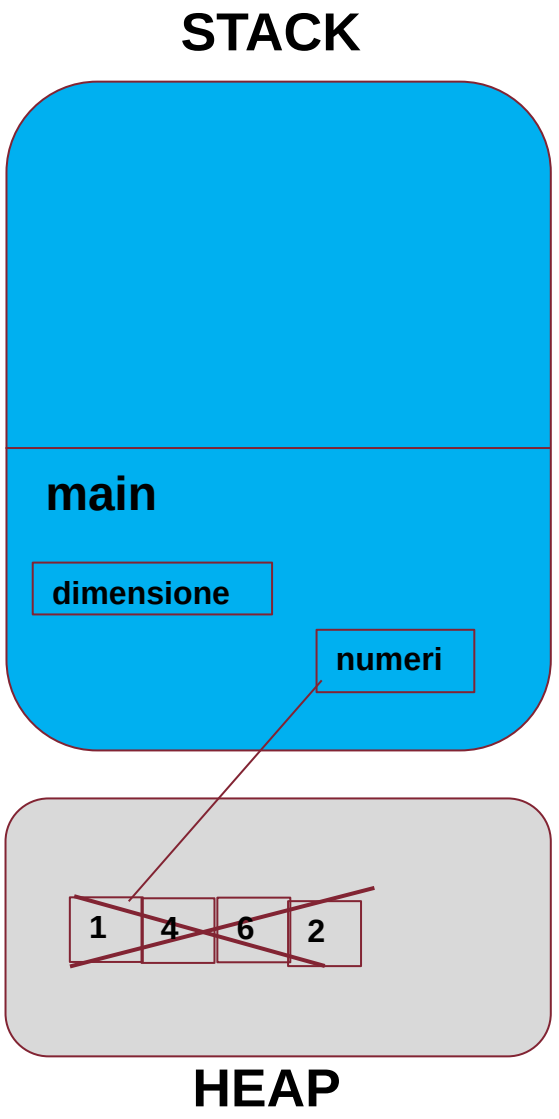
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int dimensione);
void riempi_array(int *array, int dim);
void stampa_array(int *array, int dim);

int main(){
    int dimensione = 0;
    printf("Inserire dimensione array: ");
    scanf("%d",&dimensione);
    int *numeri = crea_array(dimensione);
    if (numeri == NULL){
        printf("\nNon ci sono numeri!\n");
        return 0;
    }
    riempi_array(numeri, dimensione);
    stampa_array(numeri, dimensione);
    free(numeri);
    printf("\nMemoria liberata correttamente! \n");
    return 0;
}
```



Esercizio 2

```c
#include<stdio.h>
void funzione_1(int a, int b);
void funzione_2(int *a, int *b);
int main(){
    int a = 1;
    int b = 2;
    printf("\nMain 1 - a: %d, b: %d",a,b);
    funzione_1(a, b);
    printf("\nMain 2 - a: %d, b: %d",a,b);
    funzione_2(&a, &b);
    printf("\nMain 3 - a: %d, b: %d \n",a,b);
    return 0;
}
void funzione_1(int a, int b){
    a = b;
    b = 12;
    printf("\nfunzione_1 - a: %d, b: %d",a,b);
}
void funzione_2(int *a, int *b){
    int c = *a;
    c = 12;
    *a = *b;
    printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
}
```

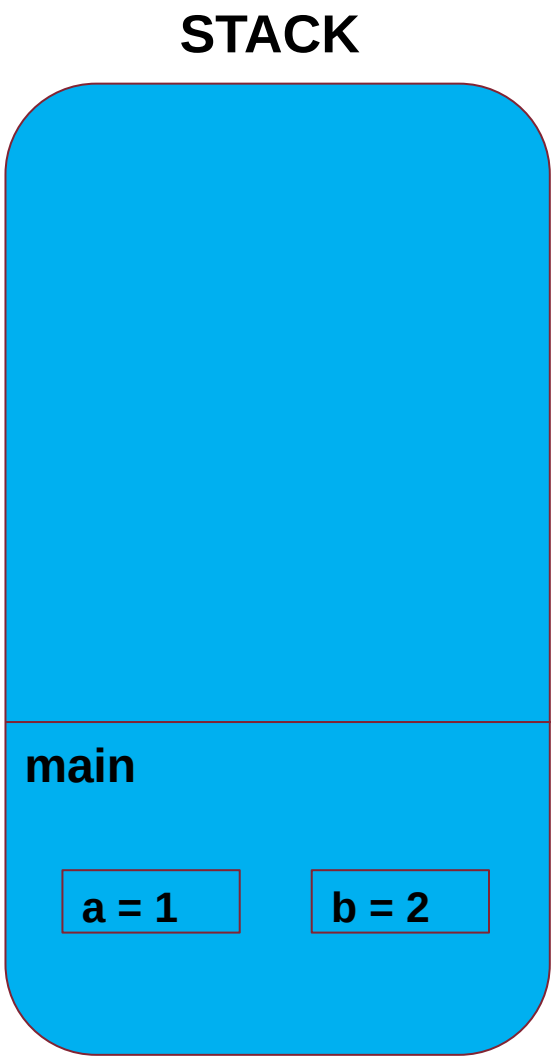
Si scriva cosa stampano le 5 printf

Esercizio 2 - Soluzione

```
Main 1 - a: 1, b: 2
funzione_1 - a: 2, b: 12
Main 2 - a: 1, b: 2
funzione_2 - a: 2, b: 2
Main 3 - a: 2, b: 2
```

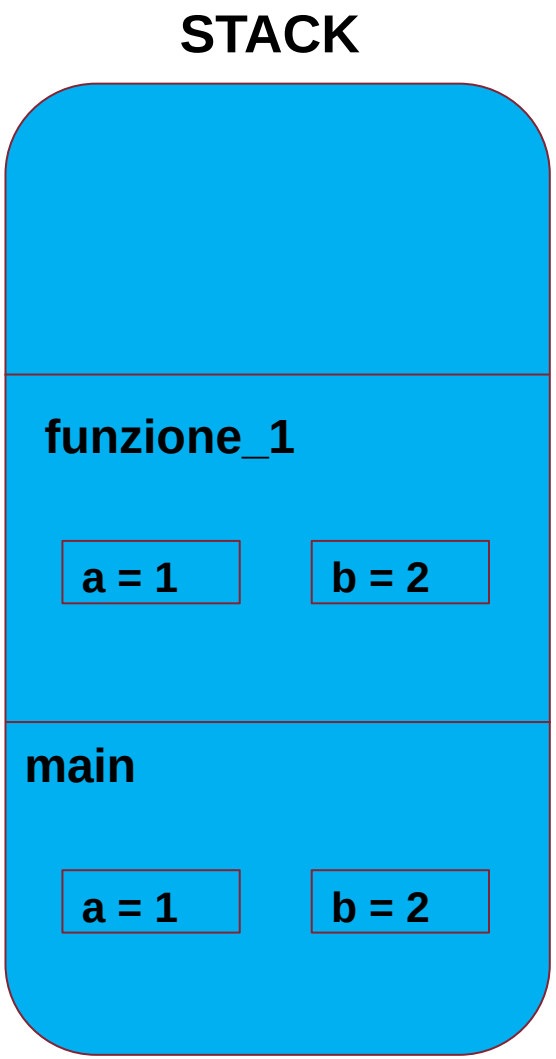
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



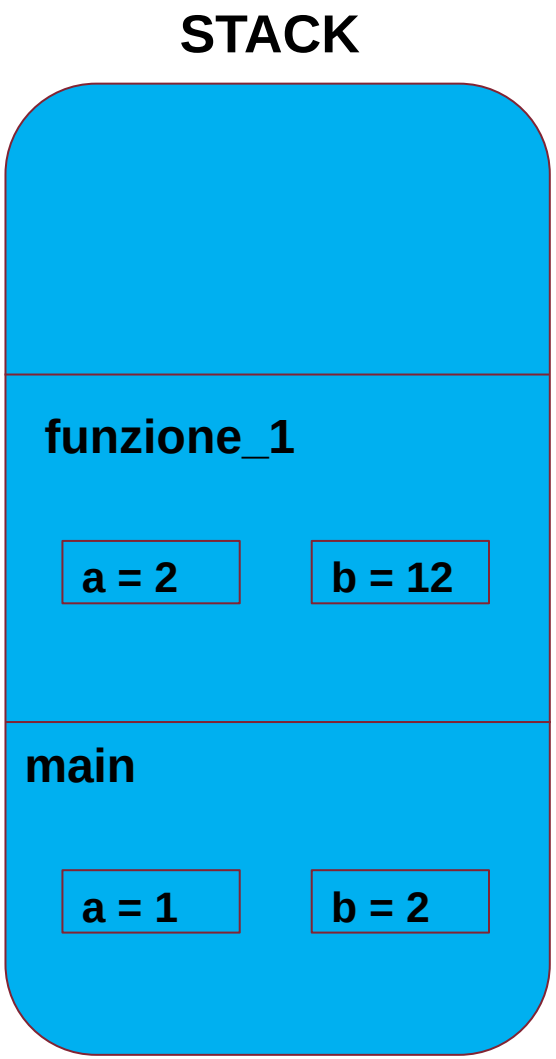
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



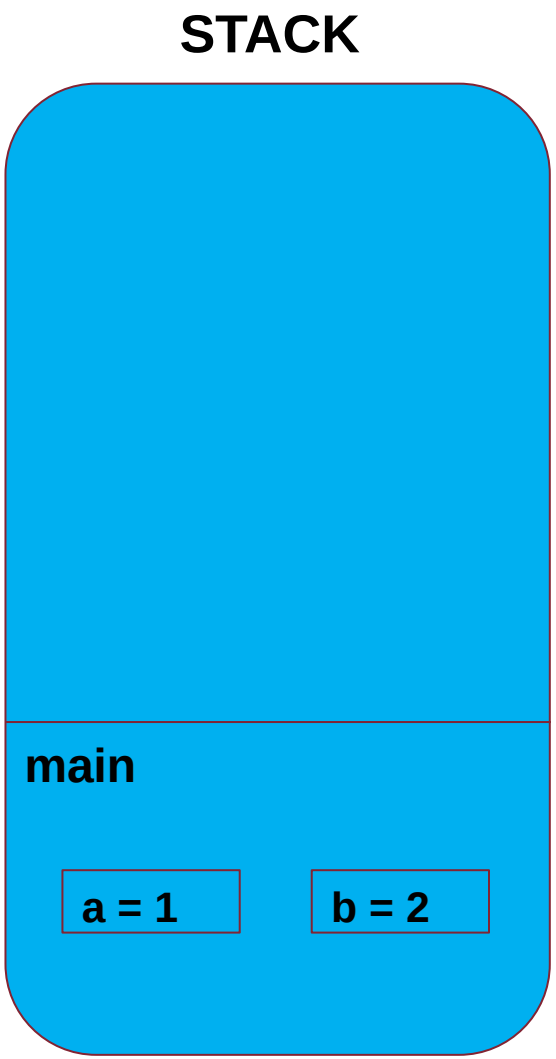
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



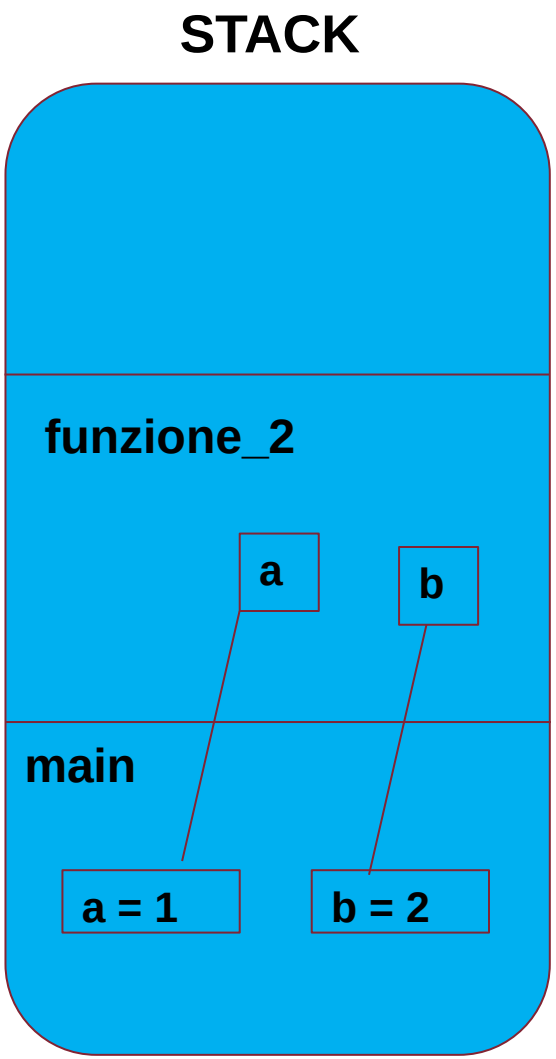
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



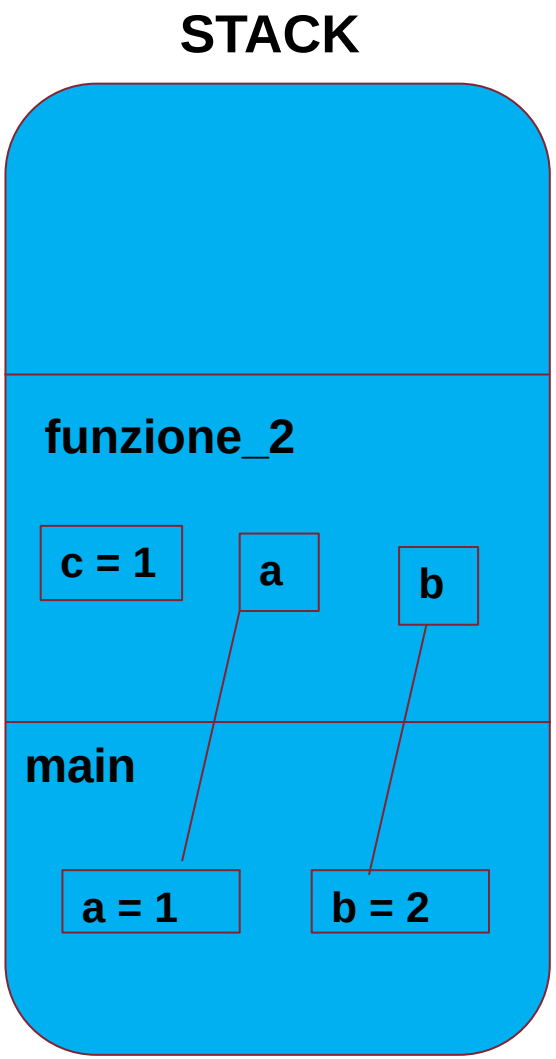
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



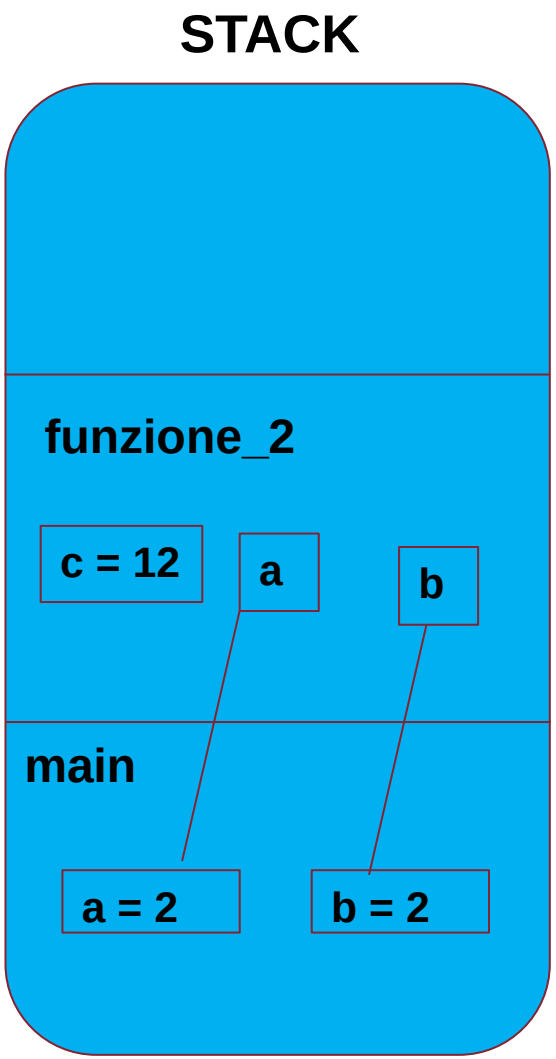
Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
void funzione_1(int a, int b);
void funzione_2(int *a, int *b);
int main(){
    int a = 1;
    int b = 2;
    printf("\nMain 1 - a: %d, b: %d",a,b);
    funzione_1(a, b);
    printf("\nMain 2 - a: %d, b: %d",a,b);
    funzione_2(&a, &b);
    printf("\nMain 3 - a: %d, b: %d \n",a,b);
    return 0;
}
void funzione_1(int a, int b){
    a = b;
    b = 12;
    printf("\nfunzione_1 - a: %d, b: %d",a,b);
}
void funzione_2(int *a, int *b){
    int c = *a;
    c = 12;
    *a = *b;
    printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
}
```



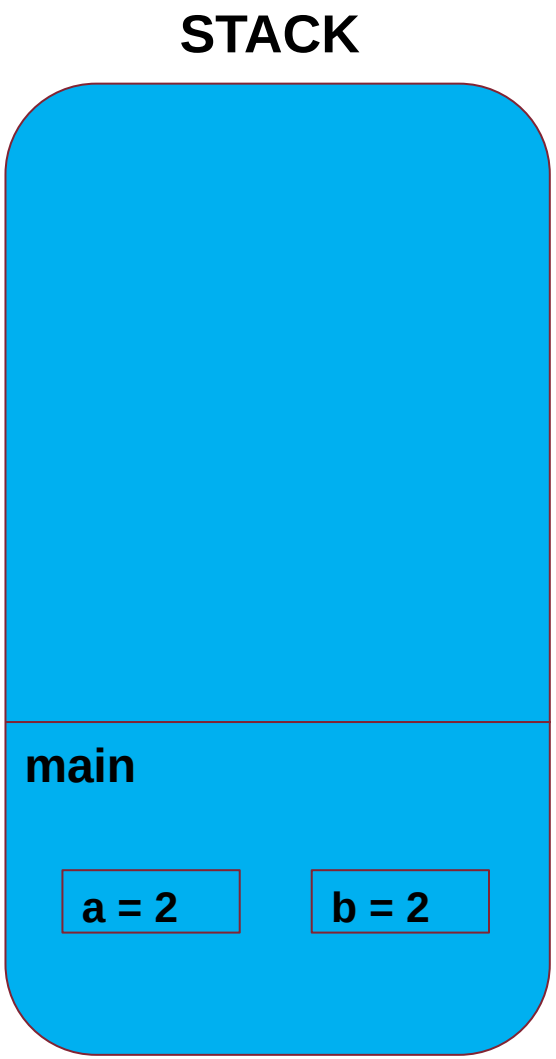
Esercizio 1 Cosa succede nella memoria...

```
1   #include<stdio.h>
2   void funzione_1(int a, int b);
3   void funzione_2(int *a, int *b);
4   int main(){
5       int a = 1;
6       int b = 2;
7       printf("\nMain 1 - a: %d, b: %d",a,b);
8       funzione_1(a, b);
9       printf("\nMain 2 - a: %d, b: %d",a,b);
10      funzione_2(&a, &b);
11      printf("\nMain 3 - a: %d, b: %d \n",a,b);
12      return 0;
13  }
14  void funzione_1(int a, int b){
15      a = b;
16      b = 12;
17      printf("\nfunzione_1 - a: %d, b: %d",a,b);
18  }
19  void funzione_2(int *a, int *b){
20      int c = *a;
21      c = 12;
22      *a = *b;
23      printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
24  }
```



Esercizio 1 Cosa succede nella memoria...

```c
#include<stdio.h>
void funzione_1(int a, int b);
void funzione_2(int *a, int *b);
int main(){
    int a = 1;
    int b = 2;
    printf("\nMain 1 - a: %d, b: %d",a,b);
    funzione_1(a, b);
    printf("\nMain 2 - a: %d, b: %d",a,b);
    funzione_2(&a, &b);
    printf("\nMain 3 - a: %d, b: %d \n",a,b);
    return 0;
}
void funzione_1(int a, int b){
    a = b;
    b = 12;
    printf("\nfunzione_1 - a: %d, b: %d",a,b);
}
void funzione_2(int *a, int *b){
    int c = *a;
    c = 12;
    *a = *b;
    printf("\nfunzione_2 - a: %d, b: %d",*a,*b);
}
```



Esercizio 3

Si scriva un programma che consenta all'utente di inserire una stringa (memorizzata in un array di 20 caratteri) e un carattere c.

Il programma deve salvare in un array di interi, allocato dinamicamente, tutte le posizioni nella stringa in cui è presente il carattere c.

Esercizio 3 - Soluzione

```c
#include<stdio.h>
#include<stdlib.h>

int *ricerca_carattere(char stringa[], char c, int *dim);

int main(){
    char chr;
    char stringa[20];
    printf("Inserire il carattere da ricercare: ");
    scanf("%c",&chr);
    printf("Inserire la stringa: ");
    scanf("%s",stringa);
    int dimensione_array = 0;
    int *array = ricerca_carattere(stringa, chr, &dimensione_array);
    printf("Posizioni in cui è presente il carattere: \n");
    for(int i=0;i<dimensione_array;i++){
        printf("%d\n",array[i]);
    }
    free(array);
}
```

Esercizio 3 - Soluzione

```
21
22  int *ricerca_carattere(char stringa[], char c, int *dim){
23      *dim = 0;
24      int *array = NULL;
25      int i = 0;
26      while(1){
27          if (stringa[i] == '\0')
28              break;
29          if (stringa[i] == c){
30              *dim = *dim + 1;
31              array = realloc(array, *dim * sizeof(int));
32              array[*dim-1] = i;
33          }
34          i = i + 1;
35      }
36      return array;
37  }
```

Esercizio 4

Si scriva un programma che consenta all'utente di inserire un numero intero.

Il programma deve memorizzare in un array di interi allocato dinamicamente, tutte le cifre del numero.

Infine si stampi a video tutte le cifre memorizzate.

Esercizio 4 - Soluzione

```c
#include<stdio.h>
#include<stdlib.h>

int *crea_array(int num, int *dimensione);

int main(){
    int numero = 0;
    printf("Inserisci un numero: ");
    scanf("%d",&numero);
    int dimensione_array = 0;
    int *array = crea_array(numero, &dimensione_array);
    for(int i=dimensione_array-1;i>=0;i--)
        printf("\n%d",array[i]);

    free(array);
    printf("\nMemoria liberata correttamente! \n");
}

int *crea_array(int num, int *dimensione){
    *dimensione = 0;
    int *array = NULL;
    while(num != 0){
        int a = num % 10;
        num = num / 10;
        *dimensione = *dimensione + 1;
        array = realloc(array, *dimensione * sizeof(int));
        array[*dimensione-1] = a;
    }
    return array;
}
```