

# Esercitazione di Reti degli elaboratori

Prof.ssa Chiara Petrioli

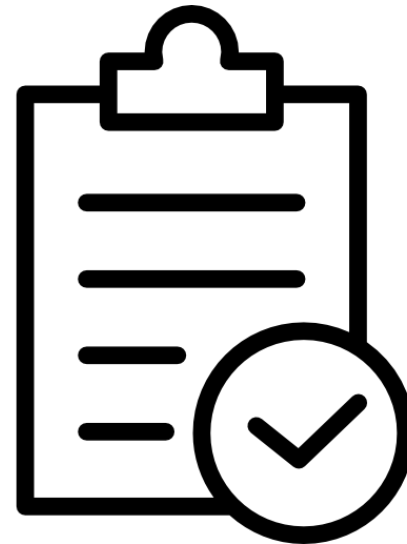


SAPIENZA  
UNIVERSITÀ DI ROMA

Corso di C

Christian Cardia, Gabriele Saturni

# Esercitazione



## Esercizio 1

Si scriva una funzione con il seguente prototipo:

*int fun(int arr[ ], int len, int \*diff)*

che prende come parametri un array di interi, la sua dimensione e un puntatore a intero. La funzione deve ritornare la somma di tutti gli elementi dell'array e deve salvare in *\*diff*, la differenza tra l'elemento maggiore ed il minore.

## Soluzione esercizio 1 (1/2)

```
11  #include<stdio.h>
12
13  //prototipo funzione
14  int fun(int arr[], int len, int *diff);
15
16  ▼ int main(){
17
18      int size = 10;
19      int array[size];
20
21  ▼    for(int i=0; i<size;i++){
22          printf("%d) Inserisci numero: ",i+1);
23          scanf("%d",&array[i]);
24          printf("\n");
25      }
26
27      int diff;
28
29      int somma = fun(array, size, &diff);
30
31      printf("\n Somma: %d, differenza: %d \n",somma,diff);
32
33      return 0;
34
35  }
```

## Soluzione esercizio 1 (2/2)

```
38 ▼ int fun(int arr[], int len, int *diff){
39
40 ▼     if(len == 0){
41         *diff = 0;
42         return 0;
43     }
44
45     int min = arr[0];
46     int max = arr[0];
47     int somma = arr[0];
48
49 ▼     for(int i=1; i<len; i++){
50         somma += arr[i];
51 ▼         if(arr[i] < min){
52             min = arr[i];
53         }
54
55 ▼         if(arr[i] > max){
56             max = arr[i];
57         }
58     }
59
60     *diff = max - min;
61     return somma;
62
63 }
```

## Esercizio 2

Si scriva una funzione con il seguente prototipo:

*int \*insertElements(int \*size)*

La funzione deve permettere all'utente di inserire numeri interi fino a che non inserisce -1, salvando i numeri su un array dinamico. La funzione deve ritornare l'indirizzo del primo elemento dell'array e salvare nel puntatore *\*size* la sua dimensione.

Dal main, dopo aver chiamato la funzione, si stampino tutti gli elementi dell'array.

## Soluzione esercizio 2 (1/2)

```
16  #include<stdio.h>
17  #include <stdlib.h>
18
19  //prototipo funzione
20  int* insertElements(int *size);
21
22  ▼ int main(){
23
24      int size = 0;
25      int *ptr = insertElements(&size);
26
27  ▼      if((size==0) || (ptr==NULL)){
28          printf("\nNessun elemento ! \n");
29          return 0;
30      }
31
32      printf("\nElementi inseriti -->");
33  ▼      for(int i=0;i<size;i++){
34          printf("\n%d)  %d",i+1,ptr[i]);
35      }
36
37      free(ptr);
38
39      printf("\n");
40
41      return 0;
42
43  }
```

## Soluzione esercizio 2 (2/2)

```
46 ▼ int* insertElements(int *size){
47
48     int *array = NULL;
49     *size = 0;
50
51     while(1){
52         int input = -1;
53         printf("\nInserire un intero oppure '-1' per terminare: ");
54         scanf("%d",&input);
55         ▼ if(input == -1){
56             break;
57         }
58         *size += 1;
59         array = (int *)realloc(array, *size * sizeof(int));
60         ▼ if(array == NULL){
61             printf("\nErrore allocazione");
62             *size = 0;
63             return array;
64         }
65         array[*size-1] = input;
66     }
67
68     return array;
69 }
```



## Esercizio 3

Si scriva cosa stampano le 5 printf:

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11     int a = 1;
12     int b = 2;
13     printf("main1 --> a: %d, b: %d",a,b);
14     funzione1(&a,b);
15     printf("\nmain2 --> a: %d, b: %d",a,b);
16     funzione2(a,&b);
17     printf("\nmain3 --> a: %d, b: %d \n",a,b);
18
19     return 0;
20 }
21
22
23
24 ▼ void funzione1(int *num1, int num2){
25     num2 = *num1 + 3;
26     *num1 = *num1 + 12;
27     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
28 }
29
30 ▼ void funzione2(int num1, int *num2){
31     *num2 = num1;
32     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
33 }
```

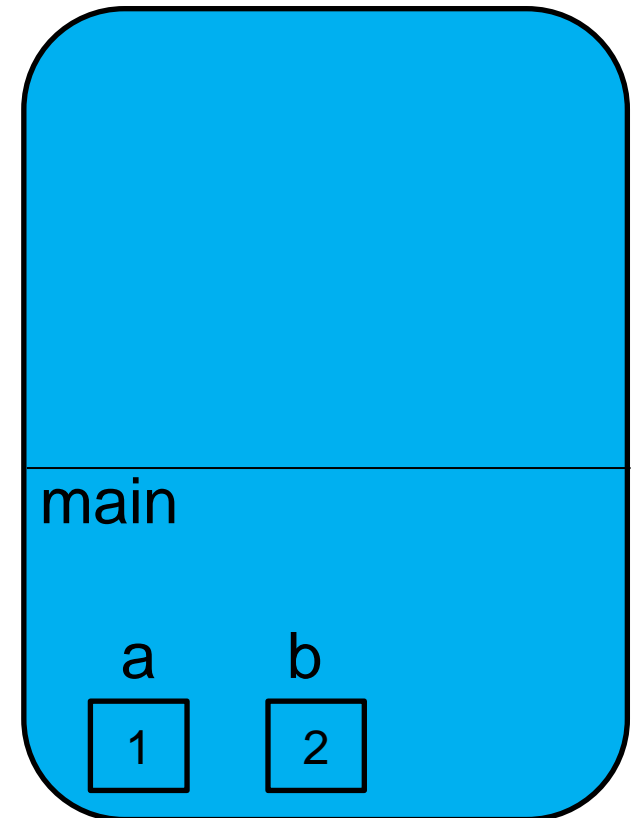
## Soluzione esercizio 3

```
main1 --> a: 1, b: 2  
funzione1 --> num1: 13, num2: 4  
main2 --> a: 13, b: 2  
funzione2 --> num1: 13, num2: 13  
main3 --> a: 13, b: 13
```

## Soluzione esercizio 3

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11     int a = 1;
12     int b = 2;
13     printf("main1 --> a: %d, b: %d",a,b);
14     funzione1(&a,b);
15     printf("\nmain2 --> a: %d, b: %d",a,b);
16     funzione2(a,&b);
17     printf("\nmain3 --> a: %d, b: %d \n",a,b);
18
19     return 0;
20 }
21
22
23
24 ▼ void funzione1(int *num1, int num2){
25     num2 = *num1 + 3;
26     *num1 = *num1 + 12;
27     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
28 }
29
30 ▼ void funzione2(int num1, int *num2){
31     *num2 = num1;
32     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
33 }
```

STACK

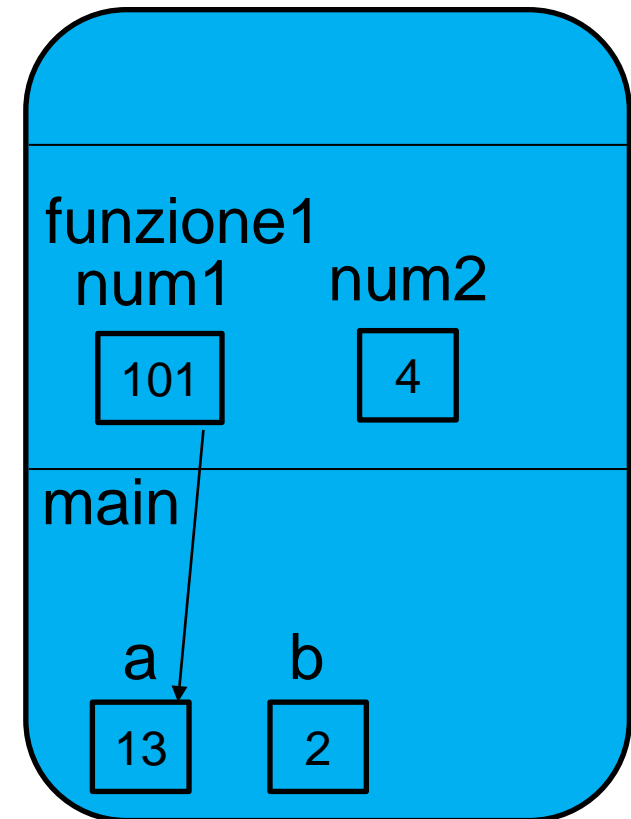


Main1 → a: 1, b: 2

## Soluzione esercizio 3

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11     int a = 1;
12     int b = 2;
13     printf("main1 --> a: %d, b: %d",a,b);
14     funzione1(&a,b);
15     printf("\nmain2 --> a: %d, b: %d",a,b);
16     funzione2(a,&b);
17     printf("\nmain3 --> a: %d, b: %d \n",a,b);
18
19     return 0;
20 }
21
22
23
24 ▼ void funzione1(int *num1, int num2){
25     num2 = *num1 + 3;
26     *num1 = *num1 + 12;
27     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
28 }
29
30 ▼ void funzione2(int num1, int *num2){
31     *num2 = num1;
32     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
33 }
```

## STACK

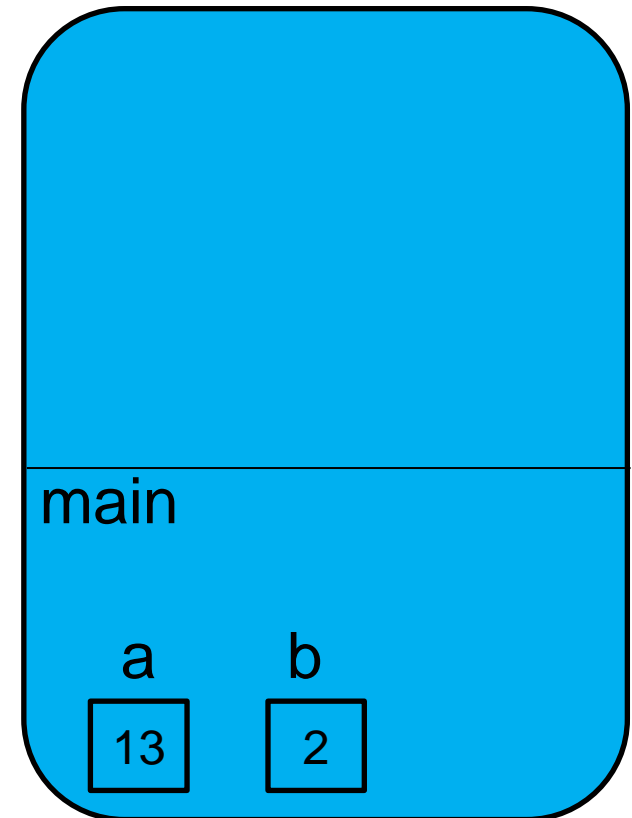


funzione1 → num1: 13, num2: 4

## Soluzione esercizio 3

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11
12     int a = 1;
13     int b = 2;
14     printf("main1 --> a: %d, b: %d",a,b);
15     funzione1(&a,b);
16     printf("\nmain2 --> a: %d, b: %d",a,b);
17     funzione2(a,&b);
18     printf("\nmain3 --> a: %d, b: %d \n",a,b);
19
20     return 0;
21
22 }
23
24 ▼ void funzione1(int *num1, int num2){
25     num2 = *num1 + 3;
26     *num1 = *num1 + 12;
27     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
28 }
29
30 ▼ void funzione2(int num1, int *num2){
31     *num2 = num1;
32     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
33 }
```

STACK

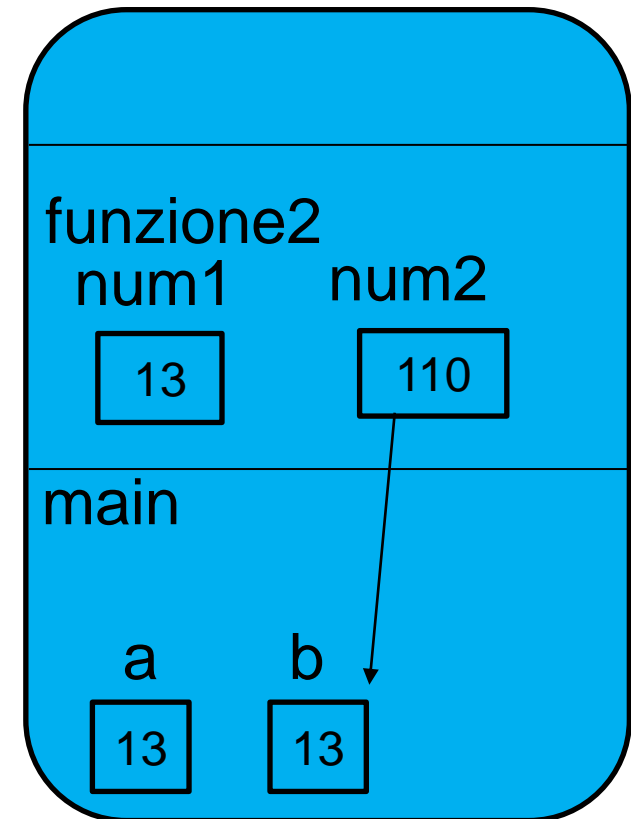


main2 → a: 13, b: 2

## Soluzione esercizio 3

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11     int a = 1;
12     int b = 2;
13     printf("main1 --> a: %d, b: %d",a,b);
14     funzione1(&a,b);
15     printf("\nmain2 --> a: %d, b: %d",a,b);
16     funzione2(a,&b);
17     printf("\nmain3 --> a: %d, b: %d \n",a,b);
18
19     return 0;
20 }
21
22 ▼ void funzione1(int *num1, int num2){
23     num2 = *num1 + 3;
24     *num1 = *num1 + 12;
25     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
26 }
27
28 ▼ void funzione2(int num1, int *num2){
29     *num2 = num1;
30     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
31 }
32
33 }
```

## STACK

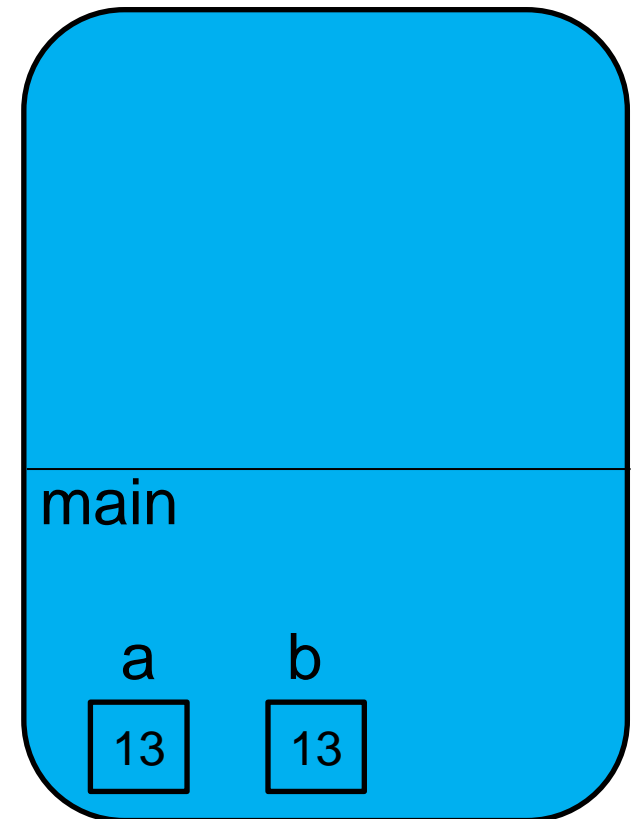


Funzione2 → num1: 13, num2: 13

## Soluzione esercizio 3

```
7 void funzione1(int *num1, int num2);
8 void funzione2(int num1, int *num2);
9
10 ▼ int main(){
11
12     int a = 1;
13     int b = 2;
14     printf("main1 --> a: %d, b: %d",a,b);
15     funzione1(&a,b);
16     printf("\nmain2 --> a: %d, b: %d",a,b);
17     funzione2(a,&b);
18     printf("\nmain3 --> a: %d, b: %d \n",a,b);
19
20     return 0;
21
22 }
23
24 ▼ void funzione1(int *num1, int num2){
25     num2 = *num1 + 3;
26     *num1 = *num1 + 12;
27     printf("\nfunzione1 --> num1: %d, num2: %d",*num1,num2);
28 }
29
30 ▼ void funzione2(int num1, int *num2){
31     *num2 = num1;
32     printf("\nfunzione2 --> num1: %d, num2: %d",num1,*num2);
33 }
```

STACK



Main3 → a: 13, b: 13

## Esercizio 4

Si scriva un programma che consenta all'utente di inserire numeri interi fino a che non ne inserisce uno negativo. Il programma deve stampare i numeri inseriti dall'utente nell'ordine inverso, la lista dei numeri pari e la lista dei numeri dispari.



## Soluzione esercizio 4 (1/3)

```
12  int* inserisciElementi(int *size);
13  void invertiNumeri(int *ptr, int size);
14  void stampaNumeri(int *ptr, int size);
15  void stampaPariDispari(int *ptr, int size);
16
17  ▼ int main(){
18
19      int size = 0;
20      int *ptr = inserisciElementi(&size);
21
22      printf("\nNumeri inseriti-->\n");
23      stampaNumeri(ptr,size);
24      printf("\n");
25      stampaPariDispari(ptr,size);
26      invertiNumeri(ptr,size);
27      printf("\nNumeri invertiti-->\n");
28      stampaNumeri(ptr,size);
29      printf("\n");
30
31      free(ptr);
32
33      return 0;
34  }
```

## Soluzione esercizio 4 (2/3)

```
36 ▼ int* inserisciElementi(int *size){
37     *size = 0;
38     int *ptr = NULL;|
39
40     while(1){
41         int input;
42         printf("\nInserisci un numero: ");
43         scanf("%d",&input);
44
45         if(input<0){
46             printf("\nFine inserimento...");
47             return ptr;
48         }
49
50         *size += 1;
51         ptr = (int *) realloc(ptr, *size * sizeof(int));
52         ptr[*size-1] = input;
53
54     }
55
56     return ptr;
57 }
58
59
60 ▼ void invertiNumeri(int *ptr, int size){
61     int a = 0;
62     for(int i=(size-1); i>(size-1)/2;i--){
63         int num = ptr[i];
64         ptr[i] = ptr[a];
65         ptr[a] = num;
66         a++;
67     }
68
69 }
```

## Soluzione esercizio 4 (3/3)

```
77 ▼ void stampaPariDispari(int *ptr, int size){
78
79     printf("\nNumeri pari-->\n");
80 ▼     for(int i=0; i<size;i++){
81 ▼         if((ptr[i] % 2)==0){
82             printf("%d-",ptr[i]);
83         }
84     }
85
86     printf("\nNumeri dispari-->\n");
87 ▼     for(int i=0; i<size;i++){
88 ▼         if((ptr[i] % 2)!=0){
89             printf("%d-",ptr[i]);
90         }
91     }
92
93 }
```

## Esercizio 5

Si scriva un programma che consenta all'utente di inserire una stringa salvata in un array dinamico. Il programma deve stampare la stringa invertita.

## Soluzione esercizio 5 (1/2)

```
7  #include<stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11  char *inserisciStringa();
12  void invertiStringa(char *stringa);
13
14  ▼ int main(){
15
16      char *stringa = inserisciStringa();
17
18      printf("\nStringa inserita: %s",stringa);
19      invertiStringa(stringa);
20      printf("\nStringa invertita: %s \n",stringa);
21
22      free(stringa);
23
24      return 0;
25  }
```

## Soluzione esercizio 5 (2/2)

```
27 ▼ char *inserisciStringa(){
28
29     int n = 0;
30
31     char ch;
32     char *stringa;
33
34     while((ch=getchar())!= '\n' ){
35         n++;
36         stringa=(char *) realloc(stringa,n+1);
37     ▼     if(stringa == NULL){
38         printf("Non è possibile allocare spazio..\n");
39         return stringa;
40     }
41     stringa[n-1]=ch;
42 }
43
44 stringa[n]= '\0';
45
46 return stringa;
47
48 }
49
50 ▼ void invertiStringa(char *stringa){
51
52     int size = strlen(stringa);
53     int a = 0;
54     ▼ for(int i=(size-1); i>((size-1)/2); i--){
55         char c = stringa[a];
56         stringa[a] = stringa[i];
57         stringa[i] = c;
58         a++;
59     }
60
61 }
```

## Esercizio 6

Si scriva cosa stampano le 5 printf:

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12     int n1 = 21;
13     int n2 = 2;
14     int *ptr;
15     ptr = &n2;
16     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
17     f1(&n1,*ptr);
18     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
19     f2(n1,&n2);
20     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
21
22     return 0;
23 }
24
25 ▼ void f1(int *a, int b){
26     *a = 32;
27     b = *a + 2;
28     printf("\nf1 --> a: %d, b: %d",*a,b);
29 }
30
31 ▼ void f2(int a, int *b){
32     *b = a + 30;
33     printf("\nf2 --> a: %d, b: %d",a,*b);
34 }
35 }
```

## Soluzione esercizio 6

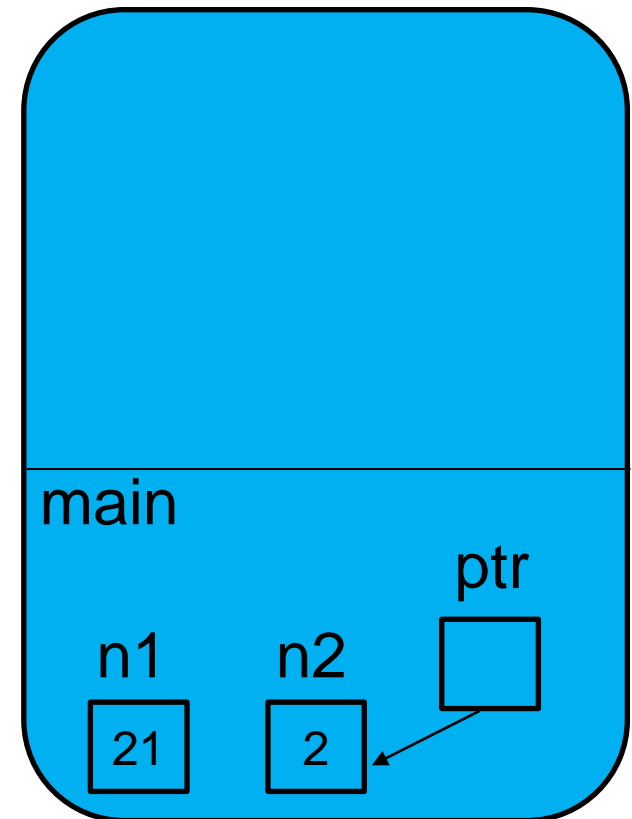
```
main1 --> n1: 21, n2: 2, ptr: 2  
f1 --> a: 32, b: 34  
main2 --> n1: 32, n2: 2, ptr: 2  
f2 --> a: 32, b: 62  
main3 --> n1: 32, n2: 62, ptr: 62
```



## Soluzione esercizio 6

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12
13     int n1 = 21;
14     int n2 = 2;
15     int *ptr;
16     ptr = &n2;
17     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
18     f1(&n1,*ptr);
19     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
20     f2(n1,&n2);
21     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
22
23     return 0;
24 }
25
26
27 ▼ void f1(int *a, int b){
28     *a = 32;
29     b = *a + 2;
30     printf("\nf1 --> a: %d, b: %d",*a,b);
31 }
32 ▼ void f2(int a, int *b){
33     *b = a + 30;
34     printf("\nf2 --> a: %d, b: %d",a,*b);
35 }
```

STACK



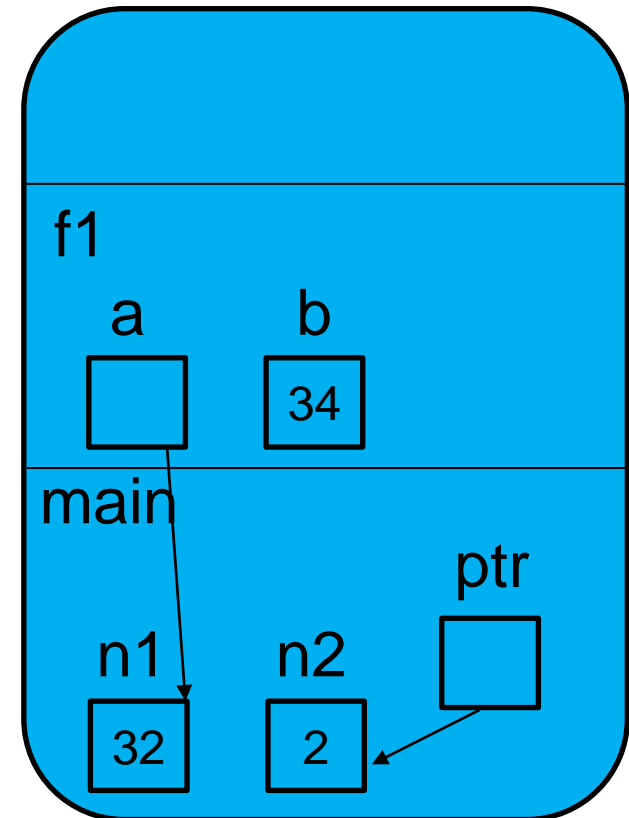
main1 → n1: 21, n2: 2, ptr: 2

## Soluzione esercizio 6

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12
13     int n1 = 21;
14     int n2 = 2;
15     int *ptr;
16     ptr = &n2;
17     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
18     f1(&n1,*ptr);
19     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
20     f2(n1,&n2);
21     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
22
23     return 0;
24 }
25
26
27 ▼ void f1(int *a, int b){
28     *a = 32;
29     b = *a + 2;
30     printf("\nf1 --> a: %d, b: %d",*a,b);
31 }
32 ▼ void f2(int a, int *b){
33     *b = a + 30;
34     printf("\nf2 --> a: %d, b: %d",a,*b);
35 }
```

f1 → a: 32, b: 34

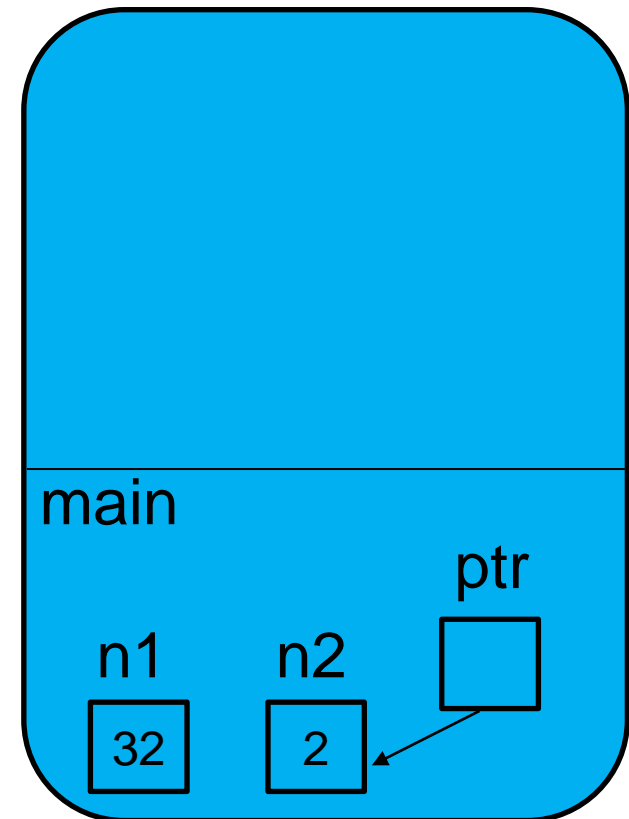
## STACK



## Soluzione esercizio 6

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12
13     int n1 = 21;
14     int n2 = 2;
15     int *ptr;
16     ptr = &n2;
17     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
18     f1(&n1,*ptr);
19     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
20     f2(n1,&n2);
21     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
22
23     return 0;
24 }
25
26
27 ▼ void f1(int *a, int b){
28     *a = 32;
29     b = *a + 2;
30     printf("\nf1 --> a: %d, b: %d",*a,b);
31 }
32 ▼ void f2(int a, int *b){
33     *b = a + 30;
34     printf("\nf2 --> a: %d, b: %d",a,*b);
35 }
```

STACK



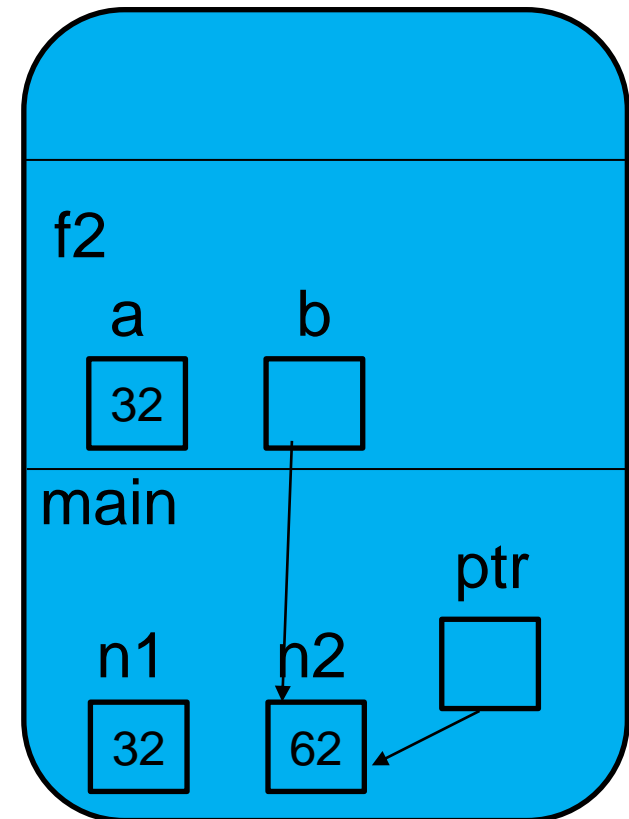
main2 → n1: 32, n2: 2, ptr: 2

## Soluzione esercizio 6

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12
13     int n1 = 21;
14     int n2 = 2;
15     int *ptr;
16     ptr = &n2;
17     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
18     f1(&n1,*ptr);
19     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
20     f2(n1,&n2);
21     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
22
23     return 0;
24 }
25
26
27 ▼ void f1(int *a, int b){
28     *a = 32;
29     b = *a + 2;
30     printf("\nf1 --> a: %d, b: %d",*a,b);
31 }
32 ▼ void f2(int a, int *b){
33     *b = a + 30;
34     printf("\nf2 --> a: %d, b: %d",a,*b);
35 }
```

f2 → a: 32, b: 62

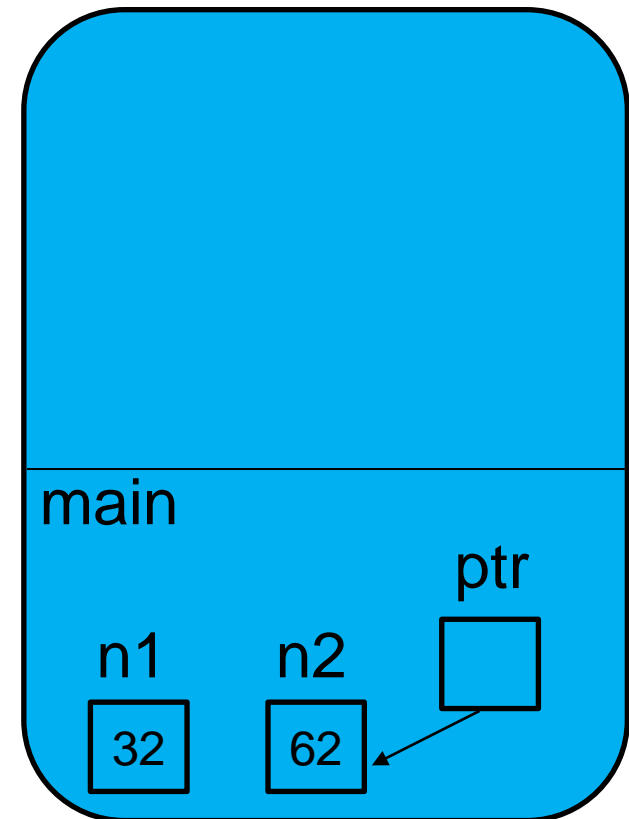
## STACK



## Soluzione esercizio 6

```
6  #include<stdio.h>
7
8  void f1(int *num1, int num2);
9  void f2(int num1, int *num2);
10
11 ▼ int main(){
12
13     int n1 = 21;
14     int n2 = 2;
15     int *ptr;
16     ptr = &n2;
17     printf("main1 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
18     f1(&n1,*ptr);
19     printf("\nmain2 --> n1: %d, n2: %d, ptr: %d",n1,n2,*ptr);
20     f2(n1,&n2);
21     printf("\nmain3 --> n1: %d, n2: %d, ptr: %d \n",n1,n2,*ptr);
22
23     return 0;
24 }
25
26
27 ▼ void f1(int *a, int b){
28     *a = 32;
29     b = *a + 2;
30     printf("\nf1 --> a: %d, b: %d",*a,b);
31 }
32 ▼ void f2(int a, int *b){
33     *b = a + 30;
34     printf("\nf2 --> a: %d, b: %d",a,*b);
35 }
```

STACK



main3 → n1: 32, n2: 62, ptr: 62

## Esercizio 7

Si scriva un programma che dichiari un array di 10 interi da far inserire all'utente. Si scriva una funzione con prototipo:

*int funzione(int arr[ ], int size, int numero, int nuovoNumero)*

La funzione deve sostituire nell'array tutte le occorrenze di «numero» con «nuovoNumero» e restituire la prima posizione in cui trova «numero». Se «numero» non è presente, deve restituire -1

## Soluzione esercizio 7 (1/2)

```
11  #include<stdio.h>
12
13  int funzione(int arr[], int size, int numero, int nuovoNumero);
14
15  ▼ int main(){
16
17      int size = 10;
18      int array[10];
19      int numeroTrovare;
20      int numeroSostituire;
21
22  ▼   for(int i=0;i<size;i++){
23       printf("\n Inserisci il numero in posizione %d: ",i+1);
24       scanf("%d",&array[i]);
25   }
26
27   printf("\nInserisci il numero da trovare: ");
28   scanf("%d",&numeroTrovare);
29   printf("\nInserisci il numero da sostituire: ");
30   scanf("%d",&numeroSostituire);
31
32   int prima_occorrenza = funzione(array, size, numeroTrovare, numeroSostituire);
33
34   printf("\nPrima occorrenza: %d",prima_occorrenza);
35
36   printf("\nArray modificato-->");
37
38  ▼   for(int i=0;i<size;i++){
39       printf("%d-",array[i]);
40   }
41
42   printf("\n");
43
44   return 0;
```

## Soluzione esercizio 7 (2/2)

```
48 ▼ int funzione(int arr[], int size, int numero, int nuovoNumero){
49
50     int prima_occorrenza = -1;
51 ▼     for(int i=0;i<size;i++){
52 ▼         if(arr[i] == numero){
53             arr[i] = nuovoNumero;
54 ▼         if(prima_occorrenza == -1){
55             prima_occorrenza = i;
56         }
57     }
58 }
59
60     return prima_occorrenza;
61 }
62
```