



Pointeri

POINTER - DECLARAREA

```
#include <stdio.h>
int main(){
    //int *p;
    //int* p;
    int * p;
    int num;
    scanf("%i", &num);
    printf("num = %i\n", num);
    printf("p = %p\n", &num);
    return 0;
}
```

ALOCAREA ADRESEI UNUI POINTER

```
#include <stdio.h>
int main(){
    int *p;
    int num;
    p = &num;
    printf("&p = %p\n", p);
    printf("&num = %p\n", &num);
    return 0;
}
```

Adresă	Valoare
A01	

CITIREA VALORII UNUI POINTER

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

```
int main(){
```

```
    int *p;
```

```
    int num;
```

```
    p = &num;
```

```
    scanf("%i", &num);
```

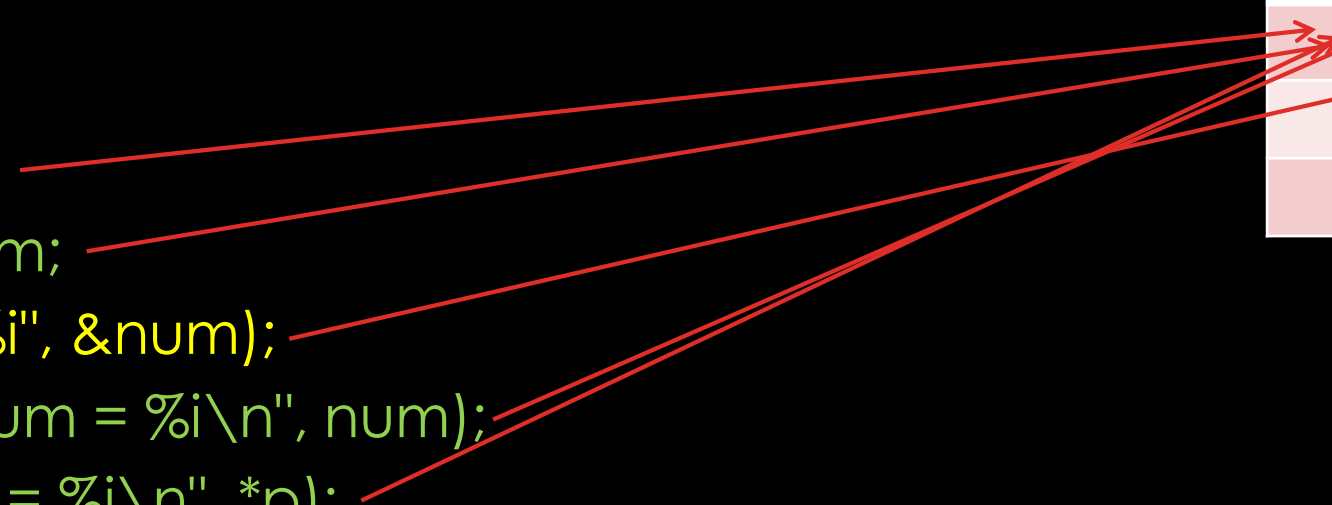
```
    printf("num = %i\n", num);
```

```
    printf("p = %i\n", *p);
```

```
    return 0;
```

```
}
```

Adresă	Valoare
A01	20



DE REȚINUT!

```
int *p;  
int a, b;
```

p, &a, &b indică adresa;

*p, a, b indică valoarea stocată în memorie.

MEMORIA CALCULATORULUI

```
#include <stdio.h>
int main(){
    int A, B;
    scanf("%i", &A);
    scanf("%i", &B);

    return 0;
}
```

*Declarare
variabile*

A

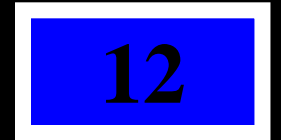


B

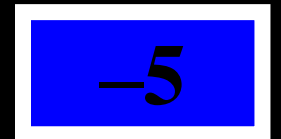


*Atribuire
valori*

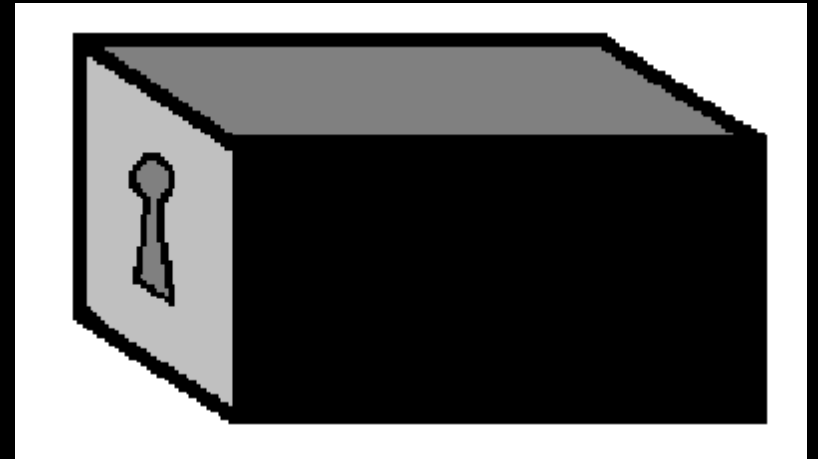
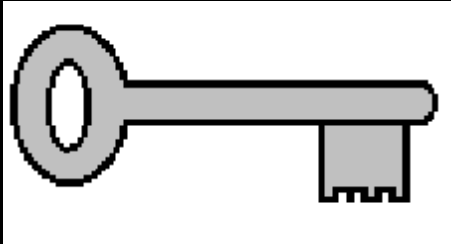
A



B



POINTER



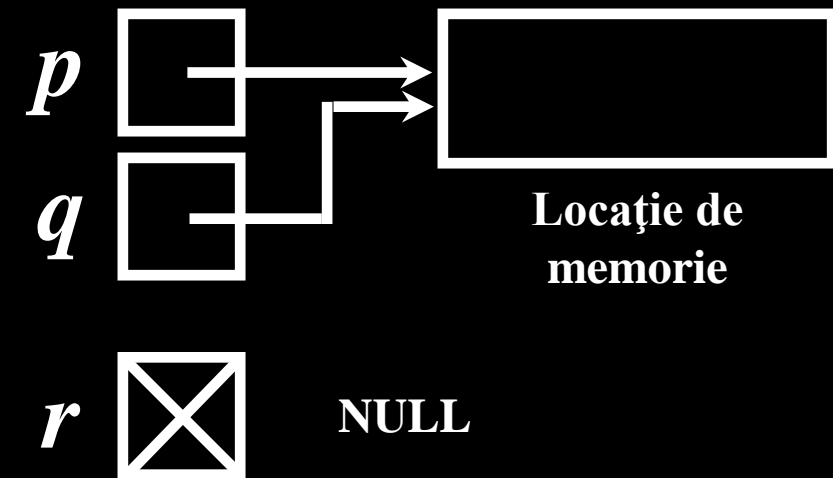
Pointer / adresă

Valoarea



POINTERI

```
#include <stdio.h>
int main(){
    int p, *q, *r;
    q = &p;
    r = NULL;
    return 0;
}
```



POINTERI

```
#include <stdio.h>
int main(){
    int r, *p, *q;
    int num;
    p = &num;
    scanf("%i", &num); // se citeste 9
    *p = 5;
    printf("num = %i\n", num);
    printf("p = %i\n", *p);
    return 0;
}
```



Ce se va
afișa?

POINTERI

```
#include <stdio.h>
int main(){
    int r = 0, *p, *q, num;
    p = &num;
    *p = 7;
    p = &r;
    printf("num = %i\n", r);
    printf("p = %i\n", *p);
    return 0;
}
```



Ce se va
afişa?

POINTERI

```
#include <stdio.h>
int main(){
    int r = 0, *p, *q, num;
    *p = 7;
    printf("num = %i\n", r);
    printf("p = %i\n", *p);
    return 0;
}
```



Ce se va
afișa?

POINTERI

```
#include <stdio.h>
int main(){
    int r = 0, *p, *q, num;
    p = &num;
    p = 2;
    q = &r;
    printf("num = %i\n", *q);
    printf("p = %i\n", *p);
    return 0;
}
```



Ce se va
afișa?



De știut!

POINTERI VECTORI

Numele unui tablou este un pointer constant spre primul element (index 0) din tablou.

- Cu alte cuvinte, o variabilă de tip tablou conține adresa de început a acestuia (adresa primei componente) și de aceea este echivalentă cu un pointer la tipul elementelor tabloului.

De Reținut!

POINTERI VECTORI

a[0]	*a
&a[0]	a
a[1]	*(a+1)
&a[1]	a+1
a[k]	*(a+k)
&a[k]	a+k

POINTERI VECTORI

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void print(int *p);
int main()
{
    int a[20];
    srand(time(0));
    for (int i = 0; i < 5; i++){
        a[i] = rand()%100;
    }
    print(a);
    return 0;
}
void print (int *p){
    for (int i = 0; i < 5; i++){
        printf("%i ", *(p+i));
    }
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void print();
int main()
{
    int a[20], *pt;
    pt = a;
    srand(time(0));
    for (int i = 0; i < 5; i++){
        a[i] = rand()%100;
    }
    print(pt);
    return 0;
}
void print (int *p){
    for (int i = 0; i < 5; i++){
        printf("%i ", *(p+i));
    }
}
```

POINTERI VECTORI

```
void swap (int * pa, int * pb) { // pointeri la intregi
int aux;
aux=*pa;
*pa=*pb;
*pb=aux; // Adresare indirecta pt a accesa valorile de la adresele pa, pb
}

// apelul acestei functii foloseşte argumente efective pointeri:
int main() {
int x=5, y=7;
swap(&x, &y); //transmitere prin adresă
printf("%d %d\n", x, y);
/*valorile sunt inversate adică se va afişa 7 5*/
return 0;
}
```




De Reținut!

POINTERI VECTORI

// Referire elemente pentru ambele variante de declarare:

v[i] sau: ***(v+i)**

int i;

double v[100], x, *p;

p=&v[0]; // corect, neelegant

p=v;

x=v[5];

x=*(v+5);

v++; // incorect

p++; //correct

REZOLVARE DE PROBLEME

Se dă un vector, elementele căruia vor fi generate aleatoriu. Să se scrie un program C, care va determina:

1. Suma elementelor pare din vector;
2. Suma elementelor divizibile cu 5;
3. Media aritmetică a elementelor de pe pozițiile impare;
4. Să se aranjeze crescător elementele vectorului folosind funcția SWAP, creată de Dvs;
5. Să se numere câte numere prime sunt în vectorul dat;
6. Să se numere câte elemente din vector sunt divizibile cu 3;
7. Să se numere câte elemente sunt mai mari ca media aritmetică a tuturor elementelor din vector;
8. Să se determine locurile elementelor egale cu X, valoare lui X este introdusă de la tastatură.

Pentru fiecare dintre cazuri se va crea o funcție. Transmiterea vectorului în funcție se va face prin pointeri.