


$$\sum_n \frac{e^{\frac{1}{\sqrt{n}}} - 1 - \frac{1}{\sqrt{n}}}{n^\alpha}$$



$$S = \sum_{i=1}^N r_n$$

sum = 0

for (int i = 0; i < N; i++) {

sum +=

}

1/ calcolo di  $r_n$

2/ calcolo  $S = \sum_n r_n$

chiedere all'utente :  $N \in [1, \dots, 10^6]$

$\alpha$  : chiedere all'utente

python: grafico di  $r_n$  in funzione di  $n$

$S$  in funzione di  $N$

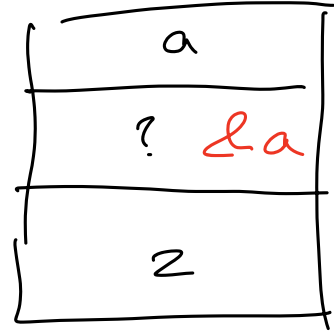
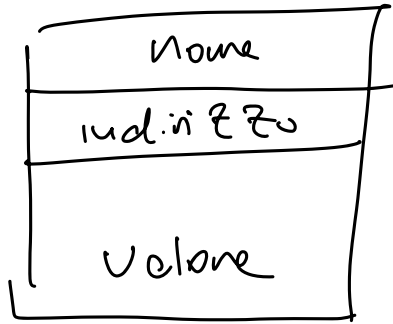
per quale converge?

Fissiamo  $N = 1000$

$\alpha = -2, -1, 1, 2, 3, \dots, 6.$

## PointerType

```
int a = 2;
```



indirizzo di a = ?

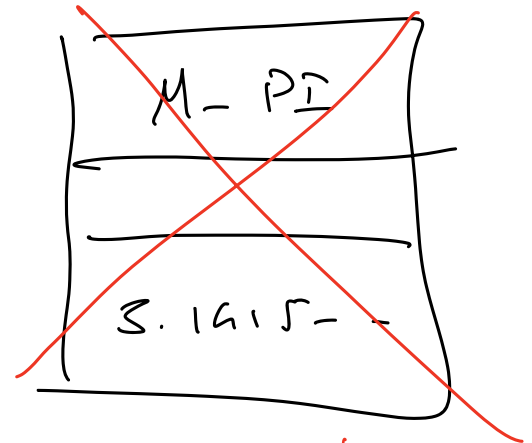
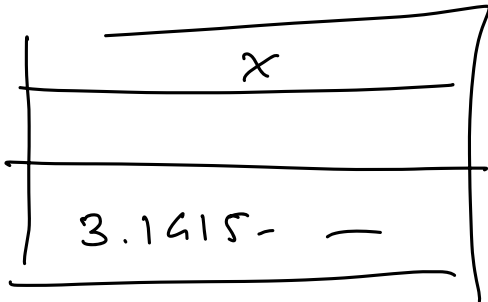
*&a;*

*&a*: indirizzo in memoria della variabile a

```
#include <math.h>
double x = M_PI;
```

*&x*

*#define M\_PI 3.1415 - -*



*non esiste*

```
printf("x = %f\n", x); // stampa valore di x
```

```
printf("&x = %p\n", &x);
```

*x = 3.1415 - -*

*&x = 0x123FABC710*

*0x123FABC710 : non  
esadecimale*

```
int* pippo;
```

*pippo è un puntatore a int.*

```
double* px;
```

*px è un puntatore a double.*

`pippo = &a;`

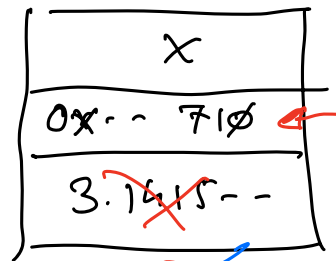
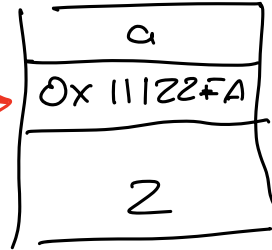
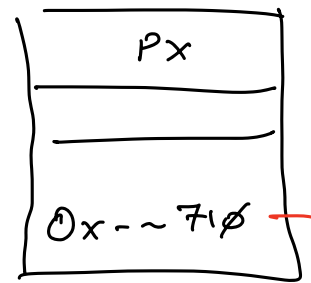
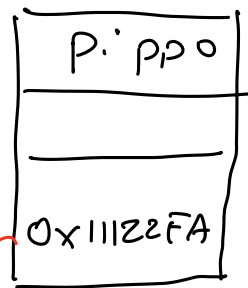
`printf("&a = %p\n", &a);`

`&a = 0x11122FA`

valore del puntatore pippo.

111

indirizzo di a



-2.2  
1.3

Come usare puntatori ?

dereferenziazione il puntatore

`*px = -2.2;`

`*px = 1.3`

`*pippo = 1024;`  $\Rightarrow$  modifica valore di a

`int b;`

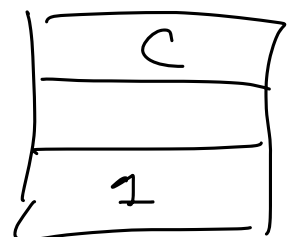
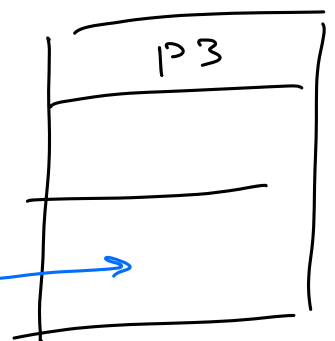
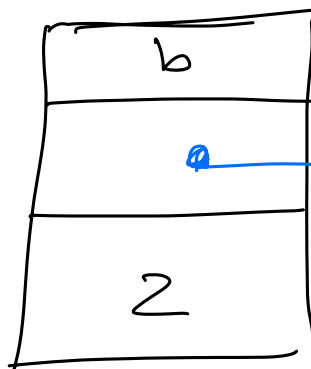
`int* p3;`

`p3 = &b`

`*p3 = 2;`

`int c = (*p3) * 2 / 4;`

`*p3` equivalente a b



```

#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define NMAX 3

int main() {

    // variabili
    double x = M_PI;
    int a = 2;

    // stampa a e x
    printf("a = %d   x = %f\n\n", a, x);

    // stampa indirizzo di a e x con il descrittore %p
    printf("&a = %p   &x = %p \n\n", &a, &x);

    // puntatore a int e double sono diversi
    int* ptr1;
    double* ptr2;

    // valore dei puntatore con %p
    printf("ptr1 = %p   ptr2 = %p \n\n", ptr1, ptr2);

    // assegnazione puntatore
    ptr1 = &a;
    ptr2 = &x;

    // valore dei puntatore con %p
    printf("ptr1 = %p   ptr2 = %p \n\n", ptr1, ptr2);

    // stampa valore di valore di *ptr1 e *ptr2
    printf("*ptr1 = %d \n", *ptr1);
    printf("*ptr2 = %f \n\n", *ptr2);

    // modifica valore di variabili a cui puntano ptr1 e ptr2
    *ptr1 = 56;
    *ptr2 = -1.2322;

    // valore dei puntatore con %p
    printf("ptr1 = %p   ptr2 = %p\n\n", ptr1, ptr2);

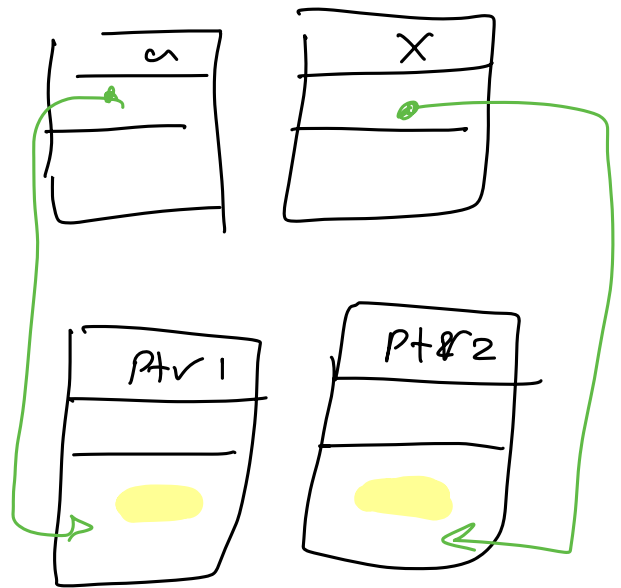
    // stampa a e x
    printf("a = %d   x = %f\n", a, x);

    // stampa valore di valore di *ptr1 e *ptr2
    printf("*ptr1 = %d \n", *ptr1);
    printf("*ptr2 = %f \n\n", *ptr2);

    // indirizzo dei puntatore
    printf("indirizzo dei puntatore:\n");
    printf("&ptr1 = %p   &ptr2 = %p\n\n", &ptr1, &ptr2);

}

```



```

shamacmini:material rahatlou$ gcc -o /tmp/app pointer.c
shamacmini:material rahatlou$ /tmp/app
a = 2   x = 3.141593

```

```

&a = 0x16fdbf284   &x = 0x16fdbf288

```

```

ptr1 = 0x201500dc0   ptr2 = 0xffffffffffffffff

```

```

ptr1 = 0x16fdbf284   ptr2 = 0x16fdbf288

```

```

*ptr1 = 2
*ptr2 = 3.141593

```

```

ptr1 = 0x16fdbf284   ptr2 = 0x16fdbf288

```

```

a = 56   x = -1.232200
*ptr1 = 56
*ptr2 = -1.232200

```

```

indirizzo dei puntatore:
&ptr1 = 0x16fdbf278   &ptr2 = 0x16fdbf270

```

$$\&ptr1 - \&ptr2 = 8$$

↓  
8 byte = 64 bit

```
int dati[3];
```

```
dati[0] = -1;
```

```
dati[1] = 2;
```

```
dati[2] = 33;
```

32 bit	32 bit	32 bit
dati[0]	dati[1]	dati[2]
-1	2	33

22

$\& \text{dati}[0]$

$\& \text{dati}[1]$

$\& \text{dati}[2]$

$\text{dati} \longleftrightarrow \&\text{dati}[0]$

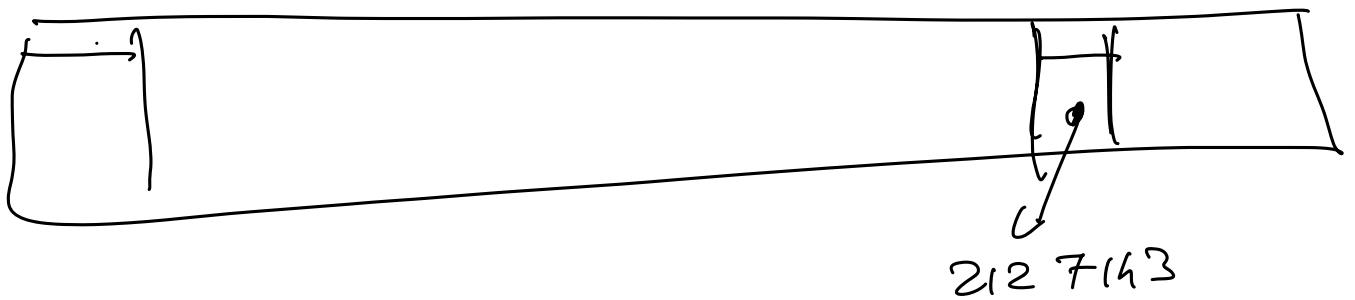
nome array    puntatore alla cella con indice 0

$\&\text{dati} = 22; \longleftrightarrow \text{dati}[0] = 22$

$\&(\text{dati} + 2) = -3 \longleftrightarrow \text{dati}[?] = -3$

`int studenti[160000];`

`studenti[122000] = 2127143;`



$\&(\text{studenti} + 122000) =$

indirizzo di memoria

`for (int i = 0; i < N; i++) {`

`dati[i] = -;`

`}`

`for (int i = 0; i < N; i++) {`

`&(\text{dati} + i) = - ,`

`}`

$(\&\text{dati}) + i \longleftrightarrow \text{dati}[0] + i$