

# Esercitazione individuale Verletta

Giovedì 8/1      } partecipano persino  
Venerdì 9/1      } con 2 assenze da ex.

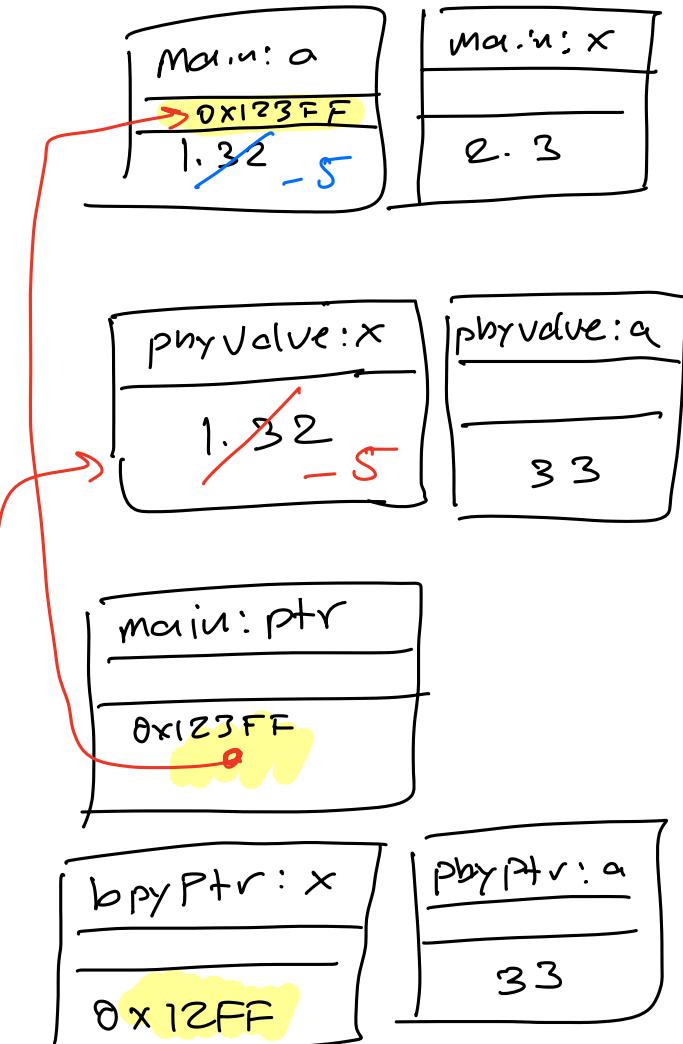
Settimana 8-12 dicembre

lun 8 festa

non c'è esercitazione: 8, 11, 12 dicembre

Passaggio per valore e per riferimento

```
1#include <stdlib.h>
2#include <stdio.h>
3
4void pbyValue(double);
5void pbyPtr(double* );
6
7int main() {
8    double a = 1.32, x = 2.3;
9    printf("main fine:\t a = %f , x = %f\n\n", a, x);
10   pbyValue(a);
11
12   printf("main dopo pbyValue: a = %f , x = %f\n\n", a, x);
13
14   double* ptr = &a;
15   pbyPtr( ptr );
16
17   printf("main fine:\t a = %f , x = %f\n\n", a, x);
18}
19
20void pbyValue(double x) {
21    double a = 33;
22    printf("inizio pbyValue: a = %f , x = %f\n", a, x);
23
24    x = -5.;
25    printf("fine pbyValue: a = %f , x = %f\n\n", a, x);
26}
27
28void pbyPtr(double* x) {
29    double a = 33;
30    printf("inizio pbyPtr: a = %f , *x = %f\n", a, *x);
31
32    *x = -5.;
33    printf("fine pbyPtr: a = %f , *x = %f\n\n", a, *x);
34}
```



$*x$  = valore delle locazioni di memoria in  $x$ .

$*x$  è 1.32

inizio pbyptr:  $a = 33$   $*x = 1.32$

$*x = 5$  : scrivi 5 nella loc. di mem  
il cui indirizzo è in x

valore di (0x123 FF) diventa -5

fine phyPtr:  $a = 33$   $*x = -5$

main fine:  $a = -5$   $x = 2.3$

```
[shamacmini:material rahatlou$ gcc -o /tmp/app passbyptr.c
```

```
[shamacmini:material rahatlou$ /tmp/app
```

main fine:  $a = 1.320000$ ,  $x = 2.300000$   
*inizio*

inizio pbyValue:  $a = 33.000000$ ,  $x = 1.320000$

fine pbyValue:  $a = 33.000000$ ,  $x = -5.000000$

main dopo pbyValue:  $a = 1.320000$ ,  $x = 2.300000$

inizio pbyPtr:  $a = 33.000000$ ,  $*x = 1.320000$

fine pbyPtr:  $a = 33.000000$ ,  $*x = -5.000000$

main fine:  $a = -5.000000$ ,  $x = 2.300000$

{ variabili locali in phyValue

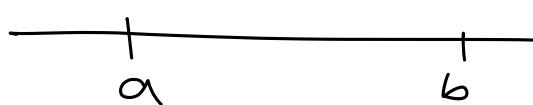
? inviate dopo phyValue

} passaggio per puntatore

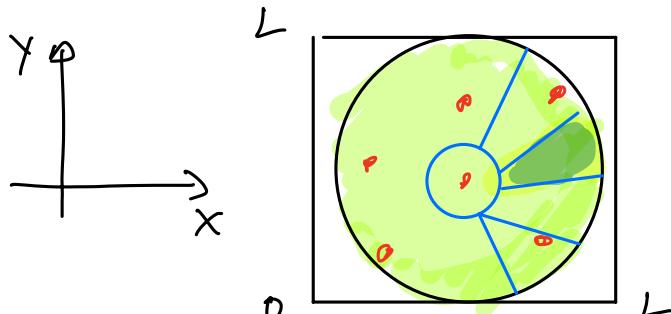
Esempio di passaggio per puntatore

double uniforme(double, double)

double x = uniforme(-2.2, 3.3);



genera uniforme  
fra  $a = -2.2$   
 $b = 3.3$ .



$$R = \sqrt{2}$$

$nCerchi = 0$ ;  
 $x = \text{uniforme}(0, L)$   
 $y = \text{uniforme}(0, L)$

// dentro o fuori?

$\text{if}((x - L/2)^2 + (y - L/2)^2 < L^2/4) \{$   
 $nCerchi++;$

}

double probCerchi = uniforme(0, 1);

$\text{if}(\text{probCerchi} \geq 0.5) \{$   
 $nCerchi++;$

generaCerchi( $\frac{L}{2}$ ,  
 $\&x, \&y$ );

do {  
     $x = \text{uniforme}(0, L);$   
     $y = \text{uniforme}(0, L);$   
} while( $(x - L/2)^2 + (y - L/2)^2 >= L^2/4$ );

$r = \sqrt{(x - L/2)^2 + (y - L/2)^2};$

$\theta = \text{atan2}(x, y)$

In C  
non si può

~~$x, y = generaCerchi(R);$~~

~~$\text{generaCerchi}(R, \&x, \&y);$~~

Funzione generaCerchi :

Dichiarazione: void generaCerchi(double, double\*,  
                          double\*);

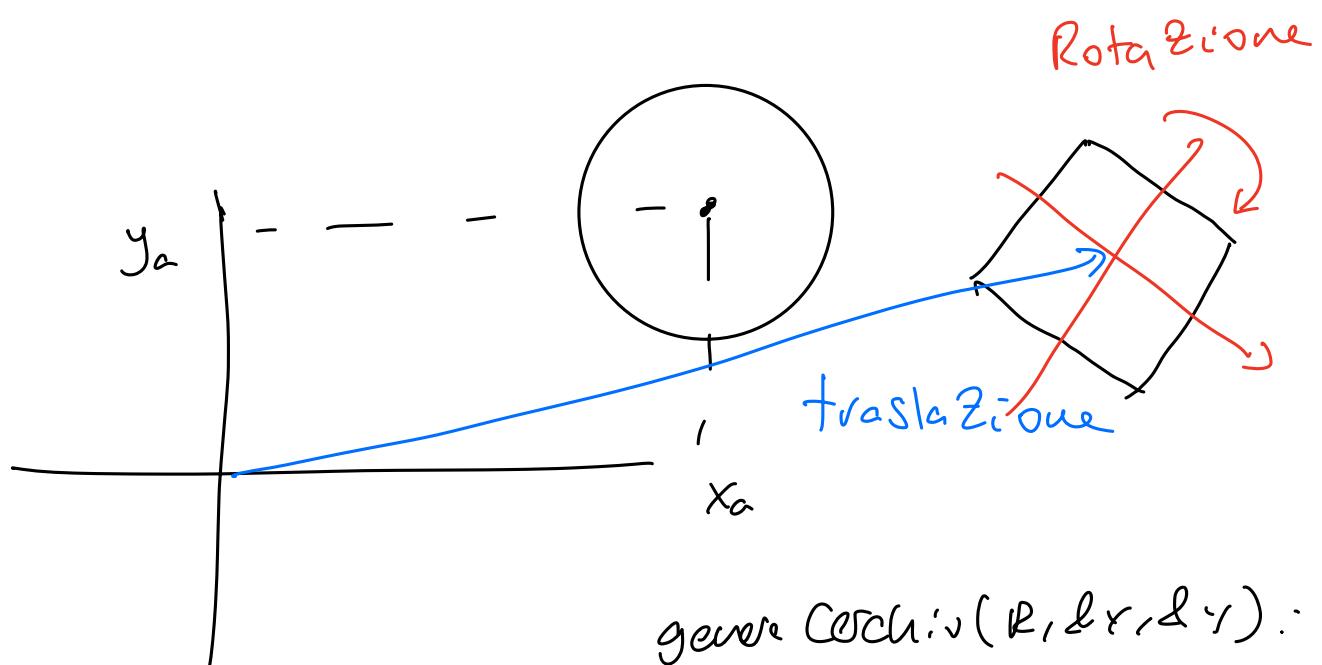
## Implementazione:

```
void generaCerchio(double R, double *x, double *y) {  
    double a, b;  
    do {  
        a = uniiforme(-R, R);  
        b = uniiforme(-R, R);  
    } while (a*a + b*b >= R*R);  
    *x = a;  
    *y = b;  
}
```

}

```
do {  
    *x = uniiforme(-R, R);  
    *y = uniiforme(-R, R);  
} while (*x * *x + *y * *y >= R*R);
```

(\*x)\*(\*x)



generaCerchio(R, dx, dy):

```
x += x_A;  
y += y_A;
```