



**POLITECNICO
DI MILANO**

INFORMATICA

L'articolazione di un
programma in
funzioni

PROGRAMMA

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  "calcola x elevato a n, con risultato in xAllaN"  
  "calcola y elevato a n, con risultato in yAllaN"  
  "calcola z elevato a n, con risultato in zAllaN"  
  if (xAllaN + yAllaN == zAllaN)
```

```
{ Leggi x,y,z,n,  
  x,y,z > 0 e n > 2
```

TEOREMA DI FERMAT

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  "calcola x elevato a n, con risultato in xAllaN"  
  "calcola y elevato a n, con risultato in yAllaN"  
  "calcola z elevato a n, con risultato in zAllaN"  
  if (xAllaN + yAllaN == zAllaN)  
    cout << "Questo caso non può presentarsi, verificare il codice"  
    << endl;  
  else  
    cout << "Il risultato conferma l'ultimo teorema di Fermat"  
    << endl;  
}
```

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  "calcola x elevato a n con risultato in xAllaN"  
  "calcola y elevato a n con risultato in yAllaN"  
  "calcola z elevato a n con risultato in zAllaN"  
  if (xAllaN + yAllaN == zAllaN)  
    cout << "Questo caso non può presentarsi, verificare il codice"  
    << endl;  
  else  
    cout << "Il risultato conferma l'ultimo teorema di Fermat"  
    << endl;  
}
```

Pseudocodice

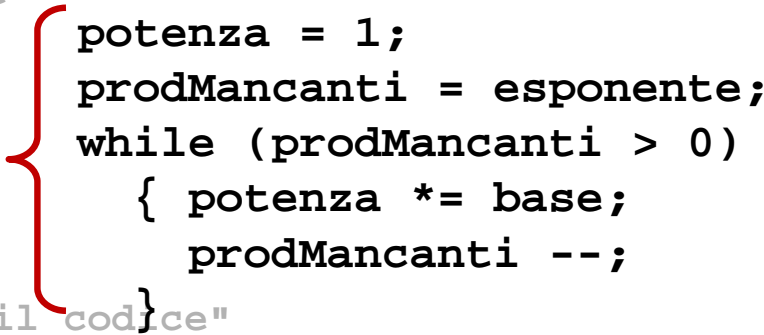
Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  "calcola x elevato a n, con risultato in xAllaN"  
  "calcola y elevato a n, con risultato in yAllaN"  
  "calcola z elevato a n, con risultato in zAllaN"  
  if (xAllaN + yAllaN == zAllaN)  
    cout << "Questo caso non può presentarsi, verificare il codice"  
    << endl;  
  else  
    cout << "Il risultato conferma l'ultimo teorema di Fermat"  
    << endl;  
}
```

Verificare se: $x^n + y^n = z^n$

```
Void main()
{...
    "leggi i dati e verifica che rispondano alle specifiche"
    "calcola x elevato a n, con risultato in xAllaN"
    "calcola y elevato a n, con risultato in yAllaN"
    "calcola z elevato a n, con risultato in zAllaN"

    if (xAllaN + yAllaN == zAllaN)
        cout << "Questo caso non può presentarsi, verificare il codice"
            << endl;
    else
        cout << "Il risultato conferma l'ultimo teorema di Fermat"
            << endl;
}
```



```
potenza = 1;
prodMancanti = esponente;
while (prodMancanti > 0)
{
    potenza *= base;
    prodMancanti --;
}
```

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
    "leggi i dati e verifica che rispondano alle specifiche"  
    //calcola x elevato a n, con risultato in xAllaN  
    base = x; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    xAllaN = potenza;  
    //calcola y elevato a n, con risultato in yAllaN  
    base = y; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    yAllaN = potenza;  
    //calcola z elevato a n, con risultato in zAllaN  
    base = z; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    zAllaN = potenza;  
    if (xAllaN + yAllaN == zAllaN)  
        cout << "Questo caso non può presentarsi, verificare il codice"  
            << endl;  
    else  
        cout << "Il risultato conferma l'ultimo teorema di Fermat"  
            << endl;  
}
```

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
    "leggi i dati e verifica che rispondano alle specifiche"  
    //calcola x elevato a n, con risultato in xAllaN  
    base = x; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    xAllaN = potenza;  
    //calcola y elevato a n, con risultato in yAllaN  
    base = y; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    yAllaN = potenza;  
    //calcola z elevato a n, con risultato in zAllaN  
    base = z; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    zAllaN = potenza;  
    if (xAllaN + yAllaN == zAllaN)  
        cout << "Questo caso non può presentarsi, verificare il codice"  
            << endl;  
    else  
        cout << "Il risultato conferma l'ultimo teorema di Fermat"  
            << endl;  
}
```


Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
    "leggi i dati e verifica che rispondano alle specifiche"  
    //calcola x elevato a n, con risultato in xAllaN  
    base = x; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    xAllaN = potenza;  
    //calcola y elevato a n, con risultato in yAllaN  
    base = y; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    yAllaN = potenza;  
    //calcola z elevato a n, con risultato in zAllaN  
    base = z; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    zAllaN = potenza;  
    if (xAllaN + yAllaN == zAllaN)  
        cout << "Questo caso non può presentarsi, verificare il codice"  
            << endl;  
    else  
        cout << "Il risultato conferma l'ultimo teorema di Fermat"  
            << endl;  
}
```

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
    "leggi i dati e verifica che rispondano alle specifiche"  
    //calcola x elevato a n, con risultato in xAllaN  
    base = x; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    xAllaN = potenza;  
    //calcola y elevato a n, con risultato in yAllaN  
    base = y; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    yAllaN = potenza;  
    //calcola z elevato a n, con risultato in zAllaN  
    base = z; esponente = n;  
    "calcola base elevato a esponente, con risultato in potenza"  
    zAllaN = potenza;  
    if (xAllaN + yAllaN == zAllaN)  
        cout << "Questo caso non può presentarsi, verificare il codice"  
            << endl;  
    else  
        cout << "Il risultato conferma l'ultimo teorema di Fermat"  
            << endl;  
}
```

BASE, ESPONENTE > 0

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  //calcola x elevato a n, con risultato in xAllaN  
  base = x; esponente = n;  
  "calcola base elevato a esponente, con risultato in potenza"  
  xAllaN = potenza;  
  //calcola y elevato a n, con risultato in yAllaN  
  base = y; esponente = n;  
  "calcola base elevato a esponente, con risultato in potenza"
```

```
...  
  potenza = 1;  
  prodMancanti = esponente;  
  while (prodMancanti > 0)  
  { potenza *= base;  
    prodMancanti --;  
  }  
  ...
```

BASE, ESPONENTE > 0

Verificare se: $x^n + y^n = z^n$

```
Void main()  
{...  
  "leggi i dati e verifica che rispondano alle specifiche"  
  //calcola x elevato a n, con risultato in xAllaN  
  base = x; esponente = n;  
  "calcola base elevato a esponente, con risultato in potenza"  
  xAllaN = potenza;  
  //calcola y elevato a n, con risultato in yAllaN  
  base = y; esponente = n;  
  "calcola base elevato a esponente, con risultato in potenza"
```



```
potenza = 1;  
for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)  
  potenza *= base;
```

```
//Tentativo senza speranza di dimostrare la falsità dell'ultimo
//teorema di Fermat

#include <iostream.h>

void main()
{ int x, y, z, n,                // valori su cui operare
  xAllaN, yAllaN, zAllaN        // contiene x,y,z, elevati a n
  int base, esponente, potenza, // variabili per utilizzo codice
  prodMancanti;                 // già esistente

//presenta le funzionalità del programma
  cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
    << "Se x, y, z sono interi positivi e n intero > 2" << endl
    << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
    << endl;

//leggi i dati e verifica che rispondano alle specifiche
  cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
  cin >> x >> y >> z >> n;
  if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

```
//calcola x elevato a n, con risultato in xAllaN
base = x; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
xAllaN = potenza;

//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
yAllaN = potenza;

//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
zAllaN = potenza;
```

```
//verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN == zAllaN)
    cout << "Risultato impossibile: Andrew Wiles ha dimostrato la"
        << "validità dell'ultimo teorema di Fermat nel 1994!"
        << endl;
else
    cout << "L'ultimo teorema di Fermat è confermato"
        << endl;
}
```

INDIPENDENZA
DELLE PARTI

```
//leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
//calcola x elevato a n, con risultato in xAllaN
base = x; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
xAllaN = potenza;
//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
yAllaN = potenza;
//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
zAllaN = potenza;
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN == zAllaN)
    cout << "Risultato impossibile: Andrew Wiles ha dimostrato la"
        << " validità dell'ultimo teorema di Fermat nel 1994!"
        << endl;
```



```
<< "Se x, y, z sono interi positivi e n intero > 2" << endl  
<< "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"  
<< endl;
```

```
//leggi i dati e verifica che rispondano alle specifiche  
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "  
    << endl;  
cin >> x >> y >> z >> n;  
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;  
  
//calcola x elevato a n, con risultato in x AllaN  
base = x; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)  
    potenza *= base;  
xAllaN = potenza;  
//calcola y elevato a n, con risultato in yAllaN  
base = y; esponente = n;
```

RETURN \cong BREAK

```
    sono interi positivi e n intero > 2" << endl
    (x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
    << endl;

//leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;

//calcola x elevato a n, con risultato in x AllaN
base = x; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
xAllaN = potenza;
//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
```

```
void main()
```

```
{ . . .
```

```
//leggi i dati e verifica che rispondano alle specifiche  
    cout << "Inserisci x,y,z n, separati da almeno uno spazio:  
        << endl;  
    cin >> x >> y >> z >> n;  
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

```
//calcola x elevato a n, con risultato in x AllaN  
    base = x; esponente = n;  
    //calcola base elevato a esponente, con risultato in potenza  
    potenza = 1;  
    for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)  
        potenza *= base;  
    xAllaN = potenza;  
//calcola y elevato a n, con risultato in yAllaN  
    base = y; esponente = n;
```

VERSIONE GREZZA

```
//leggi i dati e verifica che rispondano alle specifiche  
cout << "Inserisci x,y,z n, separati da almeno uno spazio:  
    << endl;  
cin >> x >> y >> z >> n;  
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

Il programma termina

VERSIONE RAFFINATA

```
//leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

```
<< "Se x, y, z sono interi positivi e n intero > 2" << endl  
<< "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"  
<< endl;
```

```
//leggi i dati e verifica che rispondano alle specifiche  
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "  
    << endl;  
cin >> x >> y >> z >> n;  
while (x <= 0 || y <= 0 || z <= 0 || n < 3)  
{ cout << "Attenzione! x, y, z devono essere interi positivi,"  
    << "n maggiore di 2." << endl  
    << "Inserisci nuovamente i valori separati da spazi:"  
    << endl;  
    cin >> x >> y >> z >> n;  
}
```

```
//calcola x elevato a n, con risultato in x AllaN  
base = x; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
//leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
//calcola x elevato a n, con risultato in xAllaN
base = x; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
    potenza *= base;
xAllaN = potenza;
//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
yAllaN = potenza;
//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
zAllaN = potenza;
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN == zAllaN)
    cout << "Risultato impossibile: Andrew Wiles ha dimostrato la"
        << " validità dell'ultimo teorema di Fermat nel 1994!"
        << endl;
```

1

2

3

SOTTOPROBLEMA

C + +

```
//  
cin >> x >> y >> z >> n;  
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;  
//calcola x elevato a n, con risultato in xAllaN  
base = x; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)  
    potenza *= base;  
xAllaN = potenza;  
//calcola y elevato a n, con risultato in yAllaN  
base = y; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)  
    potenza *= base;  
yAllaN = potenza;  
//calcola z elevato a n, con risultato in zAllaN  
base = z; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)  
    potenza *= base;  
zAllaN = potenza;  
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato  
if (xAllaN + yAllaN + zAllaN)  
    cout << "Risultato impossibile: Andrew Wiles ha dimostrato la"  
        << " validità dell'ultimo teorema di Fermat nel 1994!"  
        << endl;
```

nome


```
// leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
//calcola x elevato a n, con risultato in xAllaN
base = x; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
yAllaN = potenza;
//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
zAllaN = potenza;
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN == zAllaN)
    cout << "Questo caso non può presentarsi, verificare il codice"
        << endl;
```

```
// Leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
//calcola x elevato a n, con risultato in x AllaN
```

```
void funzione()
{ potenza = 1;
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)
        potenza *= base;
}
```

```
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
yAllaN = potenza;
//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
//calcola base elevato a esponente, con risultato in potenza
potenza = 1;
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
    potenza *= base;
zAllaN = potenza;
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN + zAllaN)
    cout << "Questo caso non può presentarsi, verificare il codice"
        << endl;
```

elevaAPotenza

no alle specifiche
da almeno uno spazio: "

```
cin >> x >> y >> z >> n;  
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;  
//calcola x elevato a n, con risultato in x AllaN  
  
void elevaAPotenza()  
{ potenza = 1;  
  for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)  
    potenza *= base;  
}  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)  
  potenza *= base;  
yAllaN = potenza;  
//calcola z elevato a n, con risultato in zAllaN  
base = z; esponente = n;  
//calcola base elevato a esponente, con risultato in potenza  
potenza = 1;  
for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)  
  potenza *= base;  
zAllaN = potenza;  
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato  
if (xAllaN + yAllaN + zAllaN)  
  cout << "Questo caso non può presentarsi, verificare il codice"  
  << endl;
```

```
// Leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
//calcola x elevato a n, con risultato in x AllaN

void elevaAPotenza()
{//versione con esponente positivo
    potenza = 1;
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)
        potenza *= base;
}
    potenza *= base;
    yAllaN = potenza;
//calcola z elevato a n, con risultato in zAllaN
    base = z; esponente = n;
    //calcola base elevato a esponente, con risultato in potenza
    potenza = 1;
    for (prodMancanti = 0; prodMancanti < esponenti; prodMancanti++)
        potenza *= base;
    zAllaN = potenza;
// verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN + zAllaN)
    cout << "Questo caso non può presentarsi, verificare il codice"
        << endl;
```

```
#include <iostream.h>

void main()
{
    //presenta le funzionalità del programma
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
         << "Se x, y, z sono interi positivi e n intero > 2" << endl
         << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
         << endl;

    //leggi i dati e verifica che rispondano alle specifiche
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
         << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
    //calcola x elevato a n, con risultato in x AllaN
    base = x; esponente = n;
    //calcola base elevato a esponente, con risultato in potenza
    potenza = 1;
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
        potenza *= base;
    xAllaN = potenza;

    //calcola y elevato a n, con risultato in yAllaN
    base = y; esponente = n;
    //calcola base elevato a esponente, con risultato in potenza
```

CHIAMATA

```
#include <iostream.h>
```

```
void elevaAPotenza()
```

```
{//versione con esponente positivo
```

```
    potenza = 1;
```

```
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)
```

```
        potenza *= base;
```

```
}
```

```
void main()
```

```
{
```

```
    //presenta le funzionalità del programma
```

```
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
```

```
        << "Se x, y, z sono interi positivi e n intero > 2" << endl
```

```
        << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
```

```
        << endl;
```

```
    //leggi i dati e verifica che rispondano alle specifiche
```

```
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
```

```
        << endl;
```

```
    cin >> x >> y >> z >> n;
```

```
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

```
    //calcola x elevato a n, con risultato in x AllaN
```

```
    base = x; esponente = n;
```

```
    elevaAPotenza();
```

```
    xAllaN = potenza;
```

```
#include <iostream.h>
```

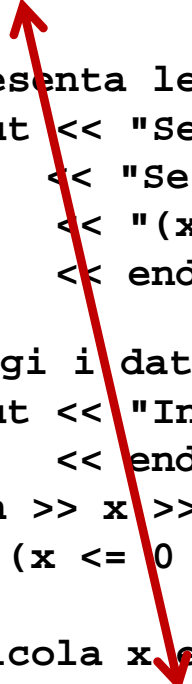
```
void elevaAPotenza()  
{//versione con esponente positivo  
    potenza = 1;  
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)  
        potenza *= base;  
}
```

```
void main()  
{  
    //presenta le funzionalità del programma  
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl  
        << "Se x, y, z sono interi positivi e n intero > 2" << endl  
        << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"  
        << endl;  
  
    //leggi i dati e verifica che rispondano alle specifiche  
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "  
        << endl;  
    cin >> x >> y >> z >> n;  
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;  
  
    //calcola x elevato a n, con risultato in x AllaN  
    base = x; esponente = n;  
    elevaAPotenza();  
    xAllaN = potenza;
```

```
#include <iostream.h>
```

```
void elevaAPotenza()  
{//versione con esponente positivo  
    potenza = 1;  
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)  
        potenza *= base;  
}
```

```
void main()  
{  
    //presenta le funzionalità del programma  
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl  
        << "Se x, y, z sono interi positivi e n intero > 2" << endl  
        << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"  
        << endl;  
  
    //leggi i dati e verifica che rispondano alle specifiche  
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "  
        << endl;  
    cin >> x >> y >> z >> n;  
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;  
  
    //calcola x elevato a n, con risultato in x AllaN  
    base = x; esponente = n;  
    elevaAPotenza();  
    xAllaN = potenza;  
}
```




```
#include <iostream.h>
```

```
void elevaAPotenza()
```

```
{// versione con esponente
```

```
    potenza = 1;
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)  
        potenza *= base;
```

```
}
```

```
void main()
```

```
{
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
        cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
```

```
//calcola x elevato a n, con risultato in x AllaN
```

```
    base = x; esponente = n;
```

```
    elevaAPotenza();
```

```
    xAllaN = potenza;
```



```
#include <iostream.h>

void miaFunzione( )
{
    //aggiunge un esponente positivo
    potenza = 1;
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
        potenza *= base;
}

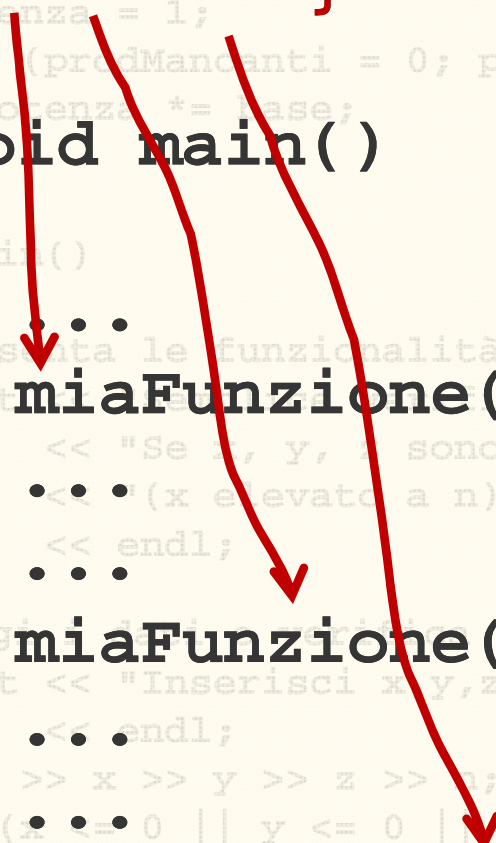
void main( )
{
    //Presenta le funzionalità del programma
    cout << "Verifica dell'ultimo teorema di Fermat." << endl
        << "Se x, y, z sono interi positivi e n intero > 2" << endl
        << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
        << endl;

    //Leggi i dati e verifica che rispondano alle specifiche
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
        << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
    //calcola x elevato a n e lo salva nel risultato in x AllaN
    base = x; esponente = n;
    miaFunzione();
    AllaN = potenza;
}
```

```
#include <iostream.h>

void miaFunzione()
{
    //codice
}

void main()
{
    //Presenta le funzionalità del programma
    miaFunzione()
    cout << "L'ultimo teorema di Fermat." << endl
    << "Se x, y, z sono interi positivi e n intero > 2" << endl
    << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
    << endl;
    //Leggi i dati che rispondano alle specifiche
    miaFunzione()
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
    //calcola x elevato a n con risultato in x AllaN
    miaFunzione()
    base = x; esponente = n;
    evaAPotenza();
    AllaN = potenza;
```



```
#include <iostream.h>

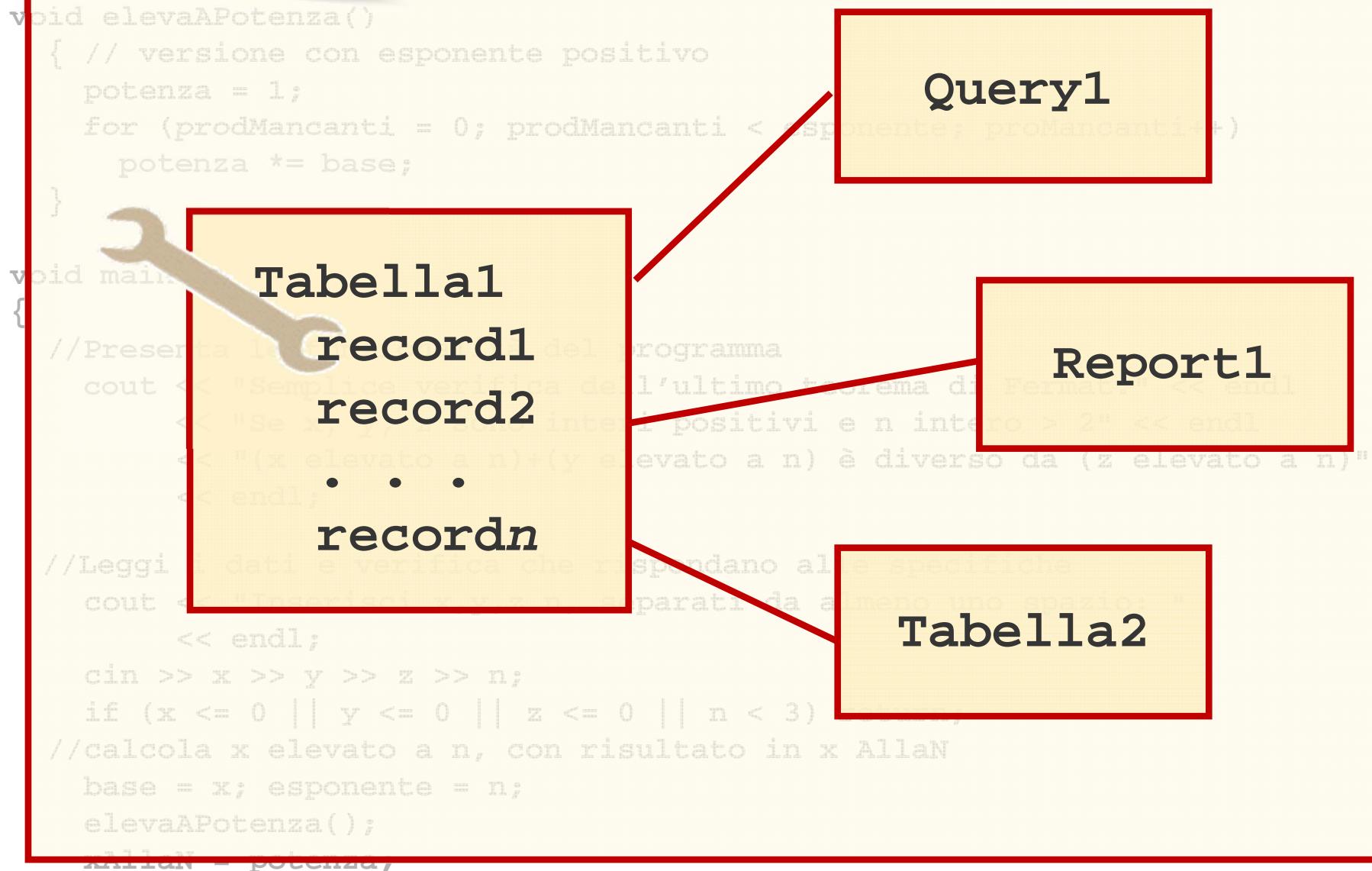
void miaFunzione()
{
    // codice
    positivo
    potenza = 1;
    for (prodMancanti = 0; prodMancanti < esponente; prodMancanti++)
        potenza *= base;
}

void main()
{
    //Presenta le funzionalità del programma
    cout << "Verifica dell'ultimo teorema di Fermat." << endl
    << "Se x, y, z sono interi positivi e n intero > 2" << endl
    << "(x elevato a n, y elevato a n) è diverso da (z elevato a n)"
    << endl;

    //Legge i dati e verifica che rispondano alle specifiche
    cout << "Inserisci x, y, z, n, separati da almeno uno spazio: "
    << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;

    //calcola x elevato a n, con risultato in x
    base = x; esponente = n;
    elevaAPotenza();
    AllaN = potenza;
```


DATABASE



```
/* Tentativo senza speranza di dimostrare la falsità dell'ultimo
 * teorema di Fermat
 * Si fa uso di na funzione senza parametri
 * Mancano le dichiarazioni di variabili.
 */

#include <iostream.h>

void elevaAPotenza()
{ //versione con esponente positivo
    potenza = 1;
    for (prodMancanti = esponente; prodMancanti > 0; proMancanti--)
        potenza *= base;
}

void main()
{
    //presenta le funzionalità del programma
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
         << "Se x, y, z sono interi positivi e n intero > 2" << endl
         << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
         << endl;
```

```
//leggi i dati e verifica che rispondano alle specifiche
cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
    << endl;
cin >> x >> y >> z >> n;
if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;

//calcola x elevato a n, con risultato in xAllaN
base = x; esponente = n;
elevatoAPotenza();
xAllaN = potenza;

//calcola y elevato a n, con risultato in yAllaN
base = y; esponente = n;
elevatoAPotenza();
yAllaN = potenza;

//calcola z elevato a n, con risultato in zAllaN
base = z; esponente = n;
elevatoAPotenza();
zAllaN = potenza;
```

```
//verifica se xAllaN + yAllaN = zAllaN e stampa il risultato
if (xAllaN + yAllaN == zAllaN)
    cout << "Risultato impossibile: Andrew Wiles ha dimostrato la"
        << "validità dell'ultimo teorema di Fermat nel 1994!"
        << endl;
else
    cout << "L'ultimo teorema di Fermat è confermato"
        << endl;
}
```



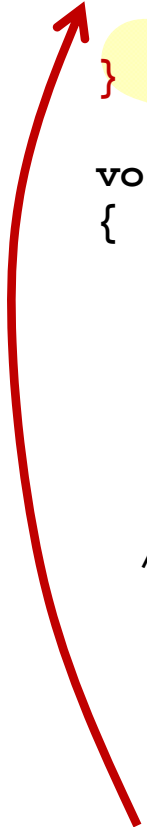
```
#include <iostream.h>

void elevaAPotenza()
{ //versione con esponente positivo
    potenza = 1;
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)
        potenza *= base;
}

void main()
{
    //presenta le funzionalità del programma
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
         << "Se x, y, z sono interi positivi e n intero > 2" << endl
         << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
         << endl;

    //leggi i dati e verifica che rispondano alle specifiche
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
         << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;

    //calcola x elevato a n, con risultato in x AllaN
    base = x; esponente = n;
    elevaAPotenza();
    xAllaN = potenza;
```



```
#include <iostream.h>
```

```
void elevaAPotenza()
```

```
{ //versione con esponente positivo
```

```
    potenza = 1;
```

```
    for (prod = 1; prod <= n; prod *= base) {  
        potenza *= base;  
    }  
}
```

```
void main()
```

```
{
```

```
    //prepara il programma
```

```
    cout << "Ultimo teorema di Fermat: n >= 3, x^n + y^n = z^n non ha soluzioni intere positive e n >= 4." << endl;
```

Dichiarazione



Uso

```
    //leggi i dati e verifica che rispondano alle specifiche
```

```
    cout << "Inserisci x,y,z n, separati da almeno uno spazio." << endl;
```

```
    << endl;
```

```
    cin >> x >> y >> z >> n;
```

```
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;
```

```
    //calcola x elevato a n, con risultato in xAllaN
```

```
    base = x; esponente = n;
```

```
    elevaAPotenza();
```

```
    xAllaN = potenza;
```

Funzione



Definizione



Uso

```
#include <iostream.h>

void elevaAPotenza()
{ //versione con esponente positivo
    potenza = 1;
    for (prodMancanti = esponente; prodMancanti > 0; prodMancanti--)
        potenza *= base;
}

void main()
{
    //presenta le funzionalità del programma
    cout << "Semplice verifica dell'ultimo teorema di Fermat." << endl
         << "Se x, y, z sono interi positivi e n intero > 2" << endl
         << "(x elevato a n)+(y elevato a n) è diverso da (z elevato a n)"
         << endl;

    //leggi i dati e verifica che rispondano alle specifiche
    cout << "Inserisci x,y,z n, separati da almeno uno spazio: "
         << endl;
    cin >> x >> y >> z >> n;
    if (x <= 0 || y <= 0 || z <= 0 || n < 3) return;

    //calcola x elevato a n, con risultato in x AllaN
    base = x; esponente = n;
    elevaAPotenza();
    xAllaN = potenza;
```