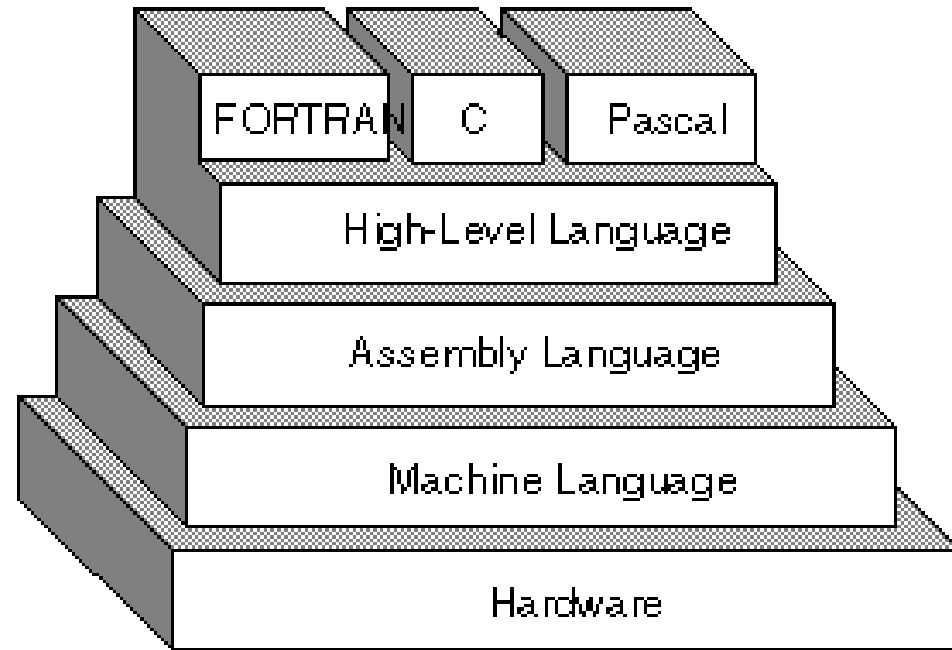


MyC

Contina Lorenzo 5EINF
2022 / 2023

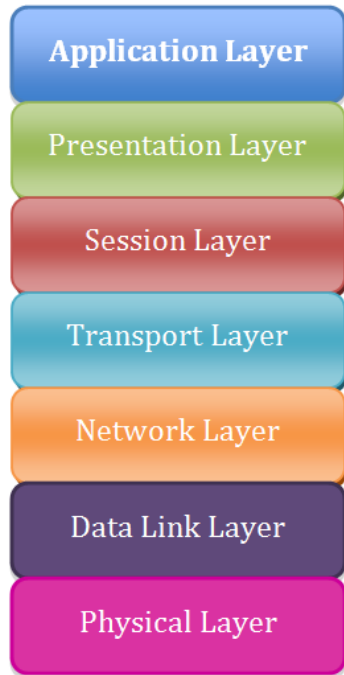
Perchè creare un nuovo linguaggio?

Allontanarsi dalla macchina

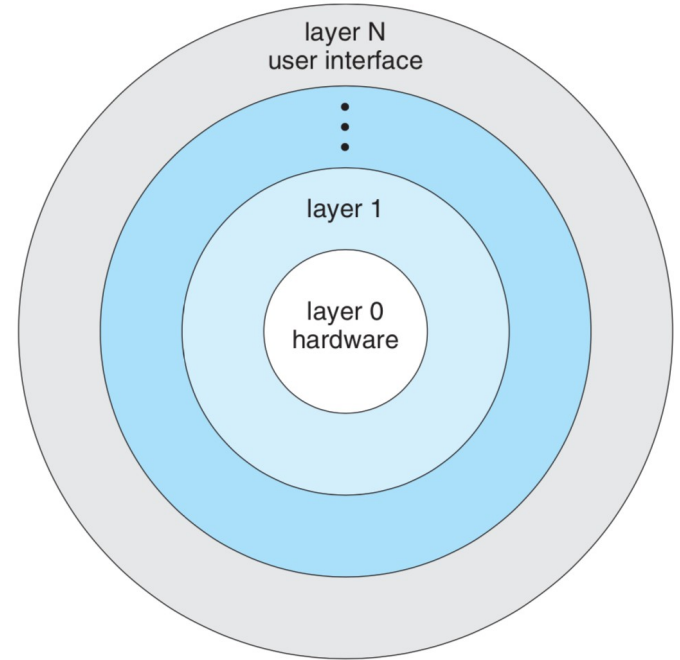
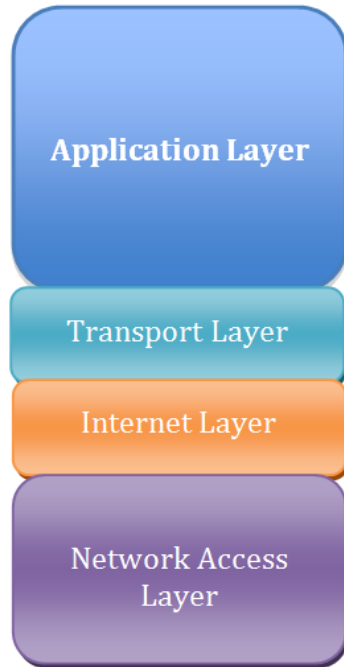


Impilare

OSI Model



TCP/IP





All'estremo

Esercizio

Creare un'applicazione che espone un set di API http che consentano ad un utente di registrarsi e gestire il proprio account (nome, cognome, data di nascita, città di origine). Inserire, utilizzando le API di GitHub, una serie di repository alla lista dei suoi preferiti. La lista deve contenere, oltre ai riferimenti per ottenere le informazioni del repository da GitHub, informazioni inserite dall'utente come: una descrizione del perché è interessato a quel repository, i linguaggi di quel repository, i tag, e la lista dei collaboratori con relativo url. I dati sono presi sia da GitHub che inseriti manualmente.

L'applicazione deve esporre delle API che consentano ad un utente loggato di elencare i propri repository preferiti, aggiungerne nuovi (cercando su GitHub) e eliminare gli esistenti, e inoltre aggiornare la descrizione. Per ogni repository devono essere inoltre visualizzate: il numero di commit/star presenti al momento della query.

■ Configuration in .NET Core

I services in .NET Core (DI)

La gestione dei dati in .NET Core

Linq

ORM (Entity Framework) + Pomelo connector

Postman

Allora dobbiamo programmare tutti
in C?

ChatGPT è il futuro della
programmazione





questa parola è un palindromo? "enna"



Sì, la parola "enna" è un palindromo. Può essere letta allo stesso modo da sinistra a destra e da destra a sinistra, mantenendo lo stesso significato.



MyC > C

Dichiarazioni

C `int *a[20];`

MyC `:a [20] *int32;`

Switch - Case

MyC

```
:type int32 = switch size {  
  case 8: {BYTE}  
  case 16: {WORD}  
  case 32: {DOUBLE}  
  case 64: {QUAD}  
};
```

C

```
int type;  
switch(size) {  
  case 8:  
    type = BYTE;  
    break;  
  case 16:  
    type = WORD;  
    break;  
  case 32:  
    type = DOUBLE;  
    break;  
  case 64:  
    type = QUAD;  
    break;  
}
```

Funzioni -- array dinamico = puntatore + dimensione

MyC

```
function max(:arr []f32) uint64 {  
    :m uint64;  
    for(:i uint64 = 0; i < arr.len; i++) {  
        if arr[i] > arr[m] {  
            m = i;  
        }  
    }  
    m  
}
```

C

```
size_t max(float *arr, size_t len) {  
    size_t m;  
    for(int i = 0; i < len; i++) {  
        if (arr[i] > arr[m]) {  
            m = i;  
        }  
    }  
    return m;  
}
```

string

MyC

```
function is_italian(:s string) bool {  
    s == "pizza"      ||  
    s == "pasta"      ||  
    s == "mandolino"  
}
```

C

```
#include <string.h>  
  
int is_italian(char* s) {  
    return strcmp(s, "pizza") ||  
        strcmp(s, "pasta") ||  
        strcmp(s, "mandolino");  
}
```

Boundary Checking

MyC

```
function max(:arr []f32) int64 {  
    :m int64;  
    for(:i int64 = 0; i < [int64]arr.len; i--) {  
        if arr[i] > arr[m] {  
            m = i;  
        }  
    }  
    m  
}
```

Risultato:

**runtime error: index -1 is
out of boundaries, with length N.
at function max
at function ...**

C

```
int max(float *arr, int len) {  
    int m;  
    for(int i = 0; i < len; i--) {  
        If (arr[i] > arr[m]) {  
            m = i;  
        }  
    }  
    return m;  
}
```

Risultato:

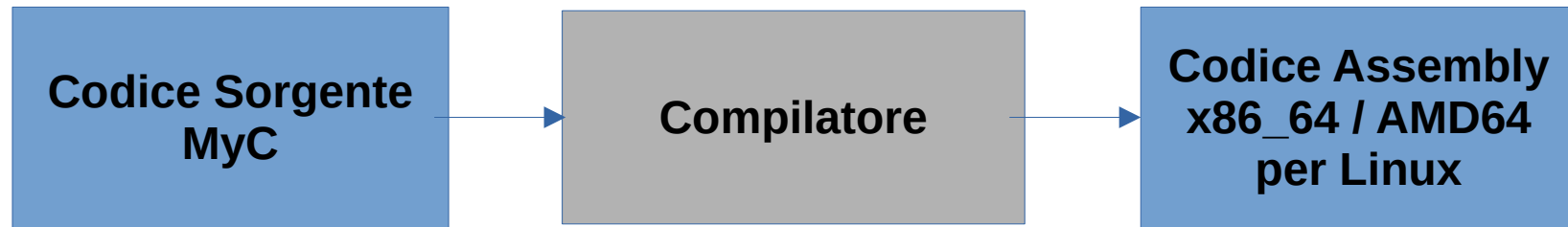
????


Alcuni dettagli implementativi

**Codice Sorgente
MyC**

Compilatore

**Codice Assembly
x86_64 / AMD64
per Linux**





```
graph LR; A["Lexer  
(analizzatore lessicale)"] -- Tokens --> B["Parser  
(analizzatore sintattico)"]; B -- AST --> C["Type System  
(tipi)"]; C -- AST --> D["Code Generator  
(generatore di codice)"]
```

Lexer
(analizzatore lessicale)

Tokens

Parser
(analizzatore sintattico)

AST

Type System
(tipi)

AST

Code Generator
(generatore di codice)

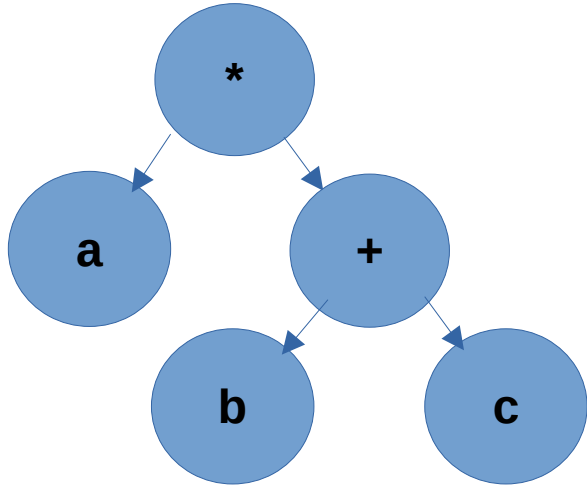
Token (simboli)

```
type Token struct {  
    Type uint32  
  
    // informazioni sulla posizione  
    // nel codice sorgente  
    L0 int32  
    C0 int32  
    L1 int32  
    C1 int32  
  
    Int_value int64  
    String_value string  
    Float_value float64  
}
```

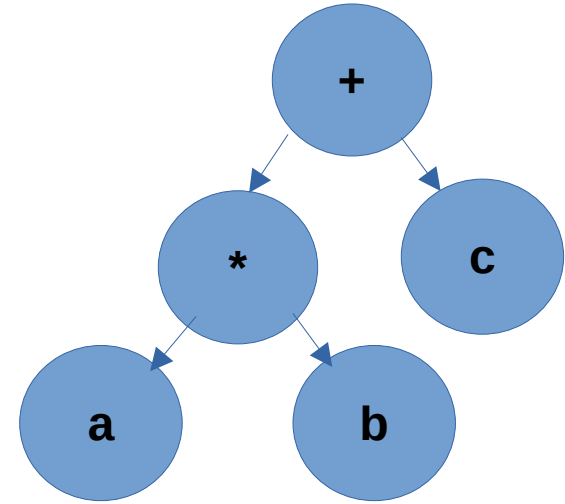
AST

```
type Ast_Node struct {  
    Type Ast_Type;  
    Flags Ast_Node_Flags;  
  
    Data []Token;  
    Children []*Ast_Node;  
  
    DataType datatype.DataType;  
}
```

Precedenza degli operatori

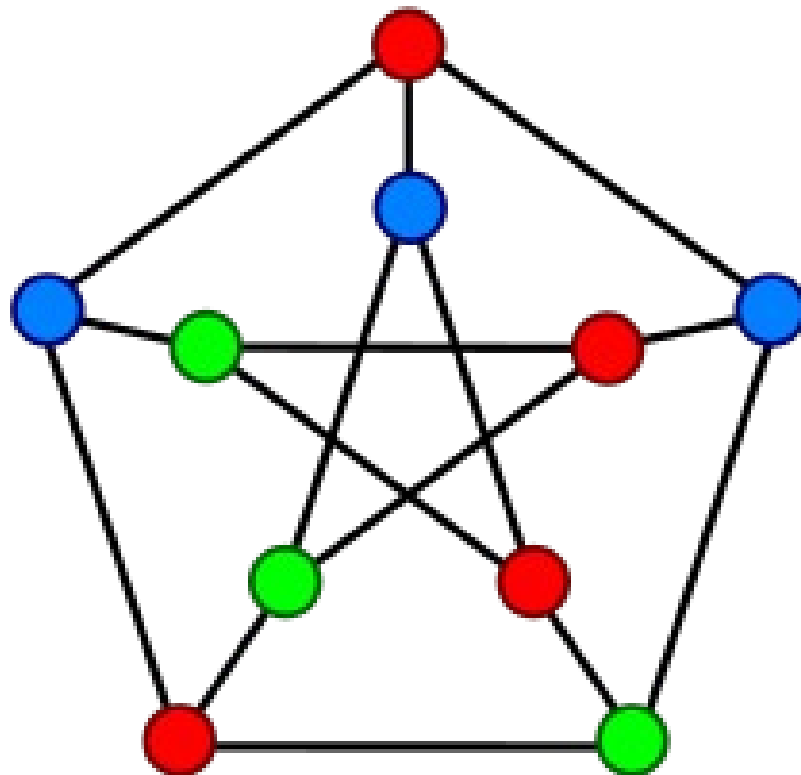


$a * b + c$



Allocazione dei registri

| x86_64 | | | |
|------------|------|------|------|
| i386 / x86 | | | |
| 8086 | | | |
| rax | eax | ax | |
| | | ah | al |
| rbx | ebx | bx | |
| | | bh | bl |
| rcx | ecx | cx | |
| | | ch | cl |
| rdx | edx | dx | |
| | | dh | dl |
| rbp | ebp | bp | bpl |
| rsl | esi | si | sll |
| rdi | edi | di | dil |
| rsp | esp | sp | spl |
| r8 | r8d | r8w | r8b |
| r9 | r9d | r9w | r9b |
| r10 | r10d | r10w | r10b |
| r11 | r11d | r11w | r11b |
| r12 | r12d | r12w | r12b |
| r13 | r13d | r13w | r13b |
| r14 | r14d | r14w | r14b |
| r15 | r15d | r15w | r15b |



Grazie!