

#####PROGETTO 24-25#####

Il progetto consiste nel sviluppare un interprete, per un linguaggio che deve soddisfare i requisiti esplicitati nella relativa sezione. Non è posto alcun vincolo aggiuntivo.

L'esame di laboratorio sarà un'orale, è richiesta la macchina personale dello studente per eseguire i programmi demo. (Qualora ci fossero delle complicazioni a riguardo, potete contattarmi via e-mail: marco.vedovato@studenti.univr.it).

La valutazione sarà in 30esimi. Il voto ottenuto sarà rapportato a 4 se consegnate il progetto prima del I appello, altrimenti a 3.

La valutazione del progetto verterà su una discussione sull'implementazione dell'interprete, e sull'esecuzione dei programmi demo. Qualora i programmi demo non funzionassero, la valutazione sarà pari a 0/30.

La prima giornata di orali per l'esame di laboratorio sarà il 20/5/25. (La prenotazione è obbligatoria).

#####REQUISITI#####

- 1) Un set di operazioni di base: +, -, /, *, %, ^, ecc. ++ per concat Str
- 2) **Non-determinismo**, es: {print(42)}ND[{print(67)}ND[print(420)]].
Es iterazioni: I) 42 II) 42 III) 67 IV) 420 V) 67 VI) 42 ... random
- 3) Ciclo **while** o **do-while**. solo il CICLO WHILE
- 4) Ciclo **for**.
- 5) Un'istruzione condizionale a scelta tra: **if**, **if-else**, **if-elif-else**, **?**. Ho scelto IF, con ELSE opzionale
- 6) Funzioni **print** e **input**.
- 7) **Stringhe**. (Consiglio di aggiungere una funzione str: Float → Stringhe).
- 8) **Float**.
- 9) **Liste** o **array**. Implementato con Mappe
- 10) **Variabili**. NON servono per forza i parametri
- 11) **Funzioni**. Senza parametri (eccezione interna per return)
- 12) Comando **sly{...}arnold**. Tale comando permette di inserire un qualunque programma Brainfuck, all'interno dei programmi che posso essere scritti col linguaggio che creerete per il progetto. (<https://copy.sh/brainfuck/> consiglio tale interprete per confortare l'output del programma brainfuck). Pertanto, l'interprete che creerete dovrà essere in grado di eseguire anche programmi scritti in Brainfuck.

#####PROGRAMMI DEMO#####

```
{ print("Buy more, more books!") } ND [ { var books = input();  
print("Buy " ++ str(books ^ 4) ++ " books" ++ "!! ;)") } ND  
[ print("Buy 42 books!") ]]
```

#####

```
fun books(){  
    print("How many books do you read in a year??");
```

```

var books = input();
while( books < ( 42 + 42 - 42 + input() ) ){
books = input();
}
ret books;
}

fun brainfuck(){
sly{
>
+ +
+ +
[ < + +
+ +
+ + + +
> - ] >
+ + + + + + + +
[
+ + + + + + + +
< - ] >
> + + > > > + >
> > > > + <
< < < < < < < <
< [ - [ - > + <
] > [ - < + > > > . < < ] > > >
[
- > + +
+ + [ >
< -
. < < [
] + > [
+ + + + + + + +
- ]
] + < <
- > + <
- < + > > > - [
[ - < < < < ] +
< ] < < < < ] + + + + + + + +
+ . + + + . [ - ] < ] + + + + + + + +
}arnold
}

```

```

print("Library → 1, Bookshop → 2.");
var books = input();
var b00ks = books();
print("Choice: " ++ str(books) ++ ", Books: " ++ str(b00ks));

print("Brainfuck: ");
brainfuck();

```

NB Quando riscriverete i programmi con il vostro linguaggio, oltre che alla semantica, tutti gli identificatori dovranno rimanere come sono stati scritti nei programmi demo.