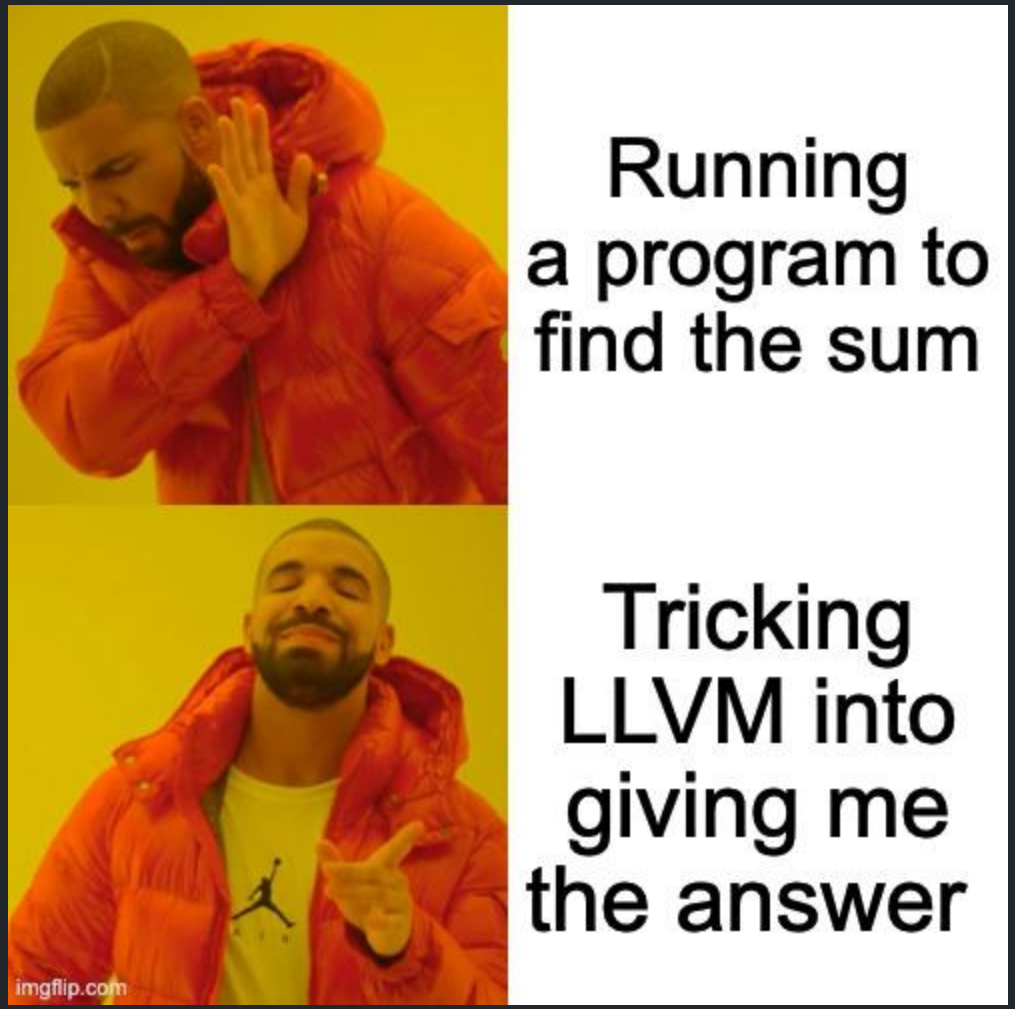
Using LLVM as a calculator

or

Just give me the .exe you smelly nerd! [1]


Running
a program to
find the sum
Tricking
LLVM into
giving me
the answer
imgflip.com

the goal

```
fn my_sum() → u32 {
    (0..1000).sum() // [0,999]
}


# has to both compile and run!
# we've got to save them clock cycles!
cargo run
```

much better

```
# only compiles, much better
cargo rustc -- --emit asm
```

arm assembly

```
_my_sum:
    sub   sp, sp, #32
    stp   x29, x30, [sp, #16]
    add   x29, sp, #16
    str   wzr, [sp, #8]
    mov   w8, #1000
    str   w8, [sp, #12]
    ldr   w0, [sp, #8]
    ldr   w1, [sp, #12]
    bl    __ZN4core4iter6traits8iterator8Iterator3sum17h8cc173ee33db684aE
    ldp   x29, x30, [sp, #16]
    add   sp, sp, #32
    ret
```

i ain’t reading all that

i’m happy for u tho

or sorry that happened

ask rustc really nicely

```
cargo rustc --release -- --emit asm
```

```
my_sum:
 mov   w0, #40748
 movk  w0, #7, lsl #16
 ret
```

more (but less) arm assembly

```
my_sum:
 mov    w0, #40748
 movk   w0, #7, lsl #16
 ;       ^   ^   ^                          7 = 0000000000000111
 ;       |   |   ├─ left shift 16 bits   7 << 16 = 1110000000000000
 ;       |   ├─ immediate 7               40748 = 0001001111100101100
 ;       ├─ dest word register           or'd = 1111001111100101100
 ret
                                    (7 << 16) | 40748
```

im not a “tech savvy computer person” so idk what that is

also why??

ARM

ope

enc

ass

the

ove

add

sep

he

n

s

i ain’t reading all that

i’m happy for u tho

or sorry that happened

ask rustc really, really nicely

```
cargo rustc \
    --target x86_64-apple-darwin \
    --release \
    -- -C opt-level=3 \
       --emit asm
```

```
my_sum:
 pushq   %rbp
 movq    %rsp, %rbp
 movl    $499500, %eax
 popq    %rbp
 retq
```

ask rustc (and mr m2) really, really, really nicely

```
cargo rustc \
    --target x86_64-unknown-linux-musl \
    --release \
    -- -C linker=x86_64-linux-musl-gcc \
       -C opt-level=3 \
       --emit asm
```

```
my_sum:
 mov  eax, 499500
 ret
```

fin

- inspired by the blog post *LLVM is Smarter Than Me*[2]
- ultimate no code solution?
- shoutouts
  - godbolt[3] the best compiler exploration tool on earth
  - our homie Gauss for his closed form summation formula
  - smart people like Krister Walfridsson who know about things like Gauss's closed form summation formula[4]

[1]: automatic_purpose_. (2024, February 17). *I am new to GitHub and I have lots to say* [Reddit Post]. r/github. www.reddit.com/r/github/comments/1at9br4/i_am_new_to_github_and_i_have_lots_to_say/

[2]: Schroer, R. (2024, April 9). *LLVM is Smarter Than Me*. Sulami's Blog. https://blog.sulami.xyz/posts/llvm-is-smarter-than-me/

[3]: Godbolt, M. (n.d.). *Compiler Explorer*. Retrieved April 22, 2024, from https://godbolt.org/

[4]: Walfridsson, K. (2019, April 28). Krister Walfridsson's old blog: How LLVM optimizes power sums. *Krister Walfridsson's Old Blog*. https://kristerw.blogspot.com/2019/04/how-llvm-optimizes-geometric-sums.html