2+4

```
mov rax, 4
mov [rbp-8], rax
mov rax, 8
mov [rbp-16], rax
mov rax, [rbp-8]
add rax, [rbp-16]
```

Registers we have used

rax, r10, r11, rsp, rdi, rbp

Access Speed (approx)

register	1 clock cycle
L1 cache	~4 clock
L2 cache	~10 clock
L3 cache	~40 clock
RAM	~100–300

Register Allocation

r12, r13, r14, r15

```
let a = 2+4 in
let b = 3-1 in
let c = a+b in
let d = b+1 in
let e = 5 in
let f = d+2 in
let g = e+f in
g
```

8 b

(1b) (4b) (5b)

```
mov rax, 4
add rax, 8
mov [rbp-8], rax
mov rax, 6
sub rax, 2
mov [rbp-16], rax
mov rax, [rbp-8]
add rax, [rbp-16]
mov [rbp-24], rax
mov rax, [rbp-16]
add rax, 2
mov [rbp-32], rax
mov rax, 10
mov [rbp-40], rax
mov rax, [rbp-32]
add rax, 4
mov [rbp-48], rax
mov rax, [rbp-40]
add rax, [rbp-48]
mov [rbp-56], rax
mov rax, [rbp-56]
```

r12, r13, r14, r15, [rbp-8]

(1) Greedy allocation algorithm

(-8, {})

[rbp-8], [rbp-16], [rbp-24], ...

(-8, [ArgRegister R12; ArgRegister R13; ...], {})

r12, r13, r14, r15, [rbp-8], ...

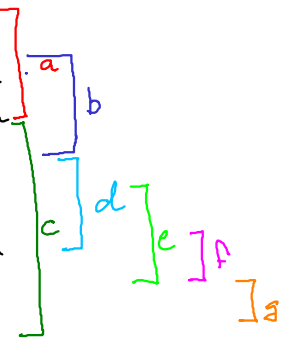
(2) Linear Scanning

* when running out of registers:
  swap some reg value into stack memory
  and re-use register

produce code ~30%-40%* slower than algs used in tools like clang

runs very fast (in compiler) — good for JIT

```
let a = 2+4 in
let b = 3-1 in
let c = a+b in
let d = b+1 in
let e = 5 in
let f = d + 2 in
let g = e + f in
g + c
```



"variable liveness"

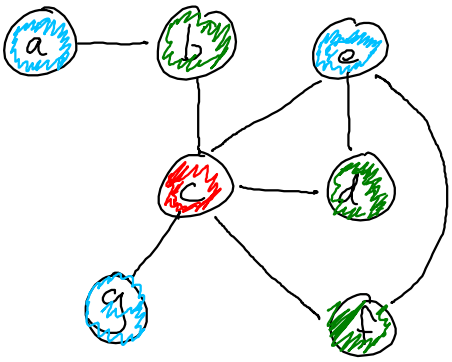
Graph Coloring

Interference Graph:

vertices : variables (conceptual storage)

edges : coexistence

color : physical storage



r12
r13
r14

Graph coloring
NP-hard

Solution ≡
find a mapping
$M : V \mapsto C$ such that

1. If $\langle V_1, V_2 \rangle \in E$, then $M(V_1) \neq M(V_2)$
2. minimal codomain of M (fewest colors possible)

Polynomial time approximation

Intermediate Representation

```
mov rax, 4
add rax, 8
mov LOC1, rax
mov rax, 6
sub rax, 2
mov LOC2, rax
mov rax, LOC1
add rax, LOC2
mov LOC3, rax
mov rax, LOC2
add rax, 2
mov LOC4, rax
mov rax, 10
mov LOC5, rax
mov rax, LOC4
add rax, 4
mov LOC6, rax
mov rax, LOC5
add rax, LOC6
mov LOC7, rax
mov rax, LOC7
```