Stage 5 Codegen Example

Tuesday, May 7, 2024 2:57 AM

```
fn main() -> int {
  let x:&int, y:int, z:&int

  entry:
    x = $alloc 10
    z = $gep x 5
    y = $load z
    y = $arith add y 1
    $store z y
    $ret y
}
```

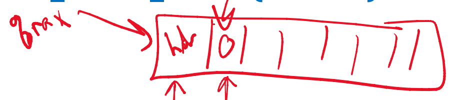
```
.data

// error messages
out_of_bounds_msg: .string "out-of-bounds array access"
invalid_alloc_msg: .string "invalid allocation amount"

.text

.globl main
main:
  pushq %rbp
  movq %rsp, %rbp
  subq $32, %rsp
  movq $0, -8(%rbp)
  movq $0, -16(%rbp)
  movq $0, -24(%rbp)
  jmp main_entry

main_entry:
  movq $10, %r8                         // check 10 > 0
  cmpq $0, %r8
  jle .invalid_alloc_length             // error if failed
  movq $1, %rdi                         // call _cflat_alloc(10 + 1)
  imulq %r8, %rdi
  incq %rdi
  call _cflat_alloc
  movq $10, %r8                         // store 10 in header
  movq %r8, 0(%rax)
  addq $8, %rax                         // store (header + <word>) in x
  movq %rax, -8(%rbp)
  movq $5, %r8                          // check 5 >= 0
  cmpq $0, %r8
  jl .out_of_bounds                     // error if failed
  movq -8(%rbp), %r9                    // check 5 < 10
  movq -8(%r9), %r10
  cmpq %r10, %r8
  jge .out_of_bounds                    // error if failed
  imulq $8, %r8                         // z = address of x[5]
  addq %r9, %r8
  movq %r8, -24(%rbp)
  movq -24(%rbp), %r8                   // y = load z
  movq 0(%r8), %r9
  movq %r9, -16(%rbp)
  movq -16(%rbp), %r8                   // y = y + 1
  addq $1, %r8
  movq %r8, -16(%rbp)
  movq -16(%rbp), %r8                   // store y into z
  movq -24(%rbp), %r9
  movq %r8, 0(%r9)
  movq -16(%rbp), %rax                  // return y
  jmp main_epilogue

main_epilogue:
  movq %rbp, %rsp
  popq %rbp
  ret

// error blocks
```

alloc

gep

load

store



```
  ret

// error blocks
.out_of_bounds:
  lea out_of_bounds_msg(%rip), %rdi
  call _cflat_panic

.invalid_alloc_length:
  lea invalid_alloc_msg(%rip), %rdi
  call _cflat_panic
```