13-1

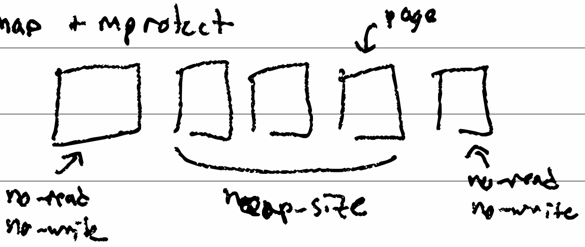
```
   cc  runtime.c   x.s    -o x.bin

X     int   main();
C     int64_t   read-int();
                                                  intptr_t
C     void  print_value ( int64_t * ty,  int64_t val);
C     int64_t*   free_ptr;
C.    int64_t*   from_space_end;
C     int64_t*   root_stack_ptr;
C     void   initialize();
C     void  collect (int64_t* root-stack_top,  int64_t alloc_reg);
   "extern"
X     int64_t    ty_unit, ty_bool, ty_s64, ty_vector;
C     char    debug;
```

```
13-3) void print_value ( *ty, val ) {
 if ( ty[0] == ty_unit) { printf("(unit)"); }
else if (ty[0] == ty_bool) {
    printf("%s", val ? "true" : "false"); }
else if ( ty[0] == ty_s64) {
    printf("%d", val); }
else if (ty[0] == ty_vector) {
    printf("(vector");
   for (int i=0; i < ty[1]; i++) {
     printf("␣");
     print_value (ty[2+i], val[1+i]); }
  printf(")"); } }
```

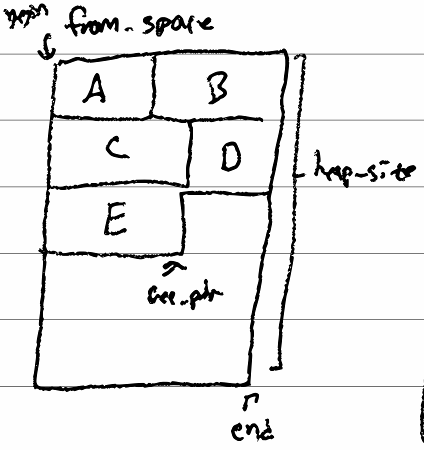
13-3) void initialize() {

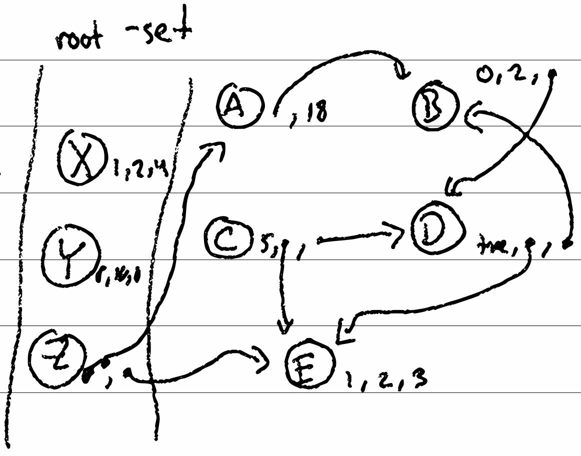
```
  from_space_begin = malloc(heap_size);
  from_space_end = from_space_begin + heap_size;
  free_ptr = from_space_begin;
  root_stack_begin = malloc(heap_size);
  root_stack_ptr = root_stack_end; }
size_t heap_size = (1 << 10);
```

mmap + mprotect



13-4 / Stop + copy

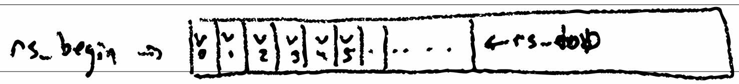




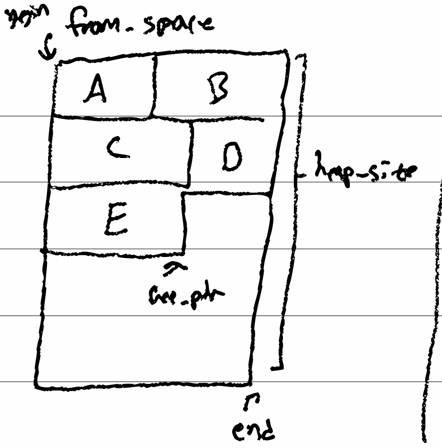
root set:

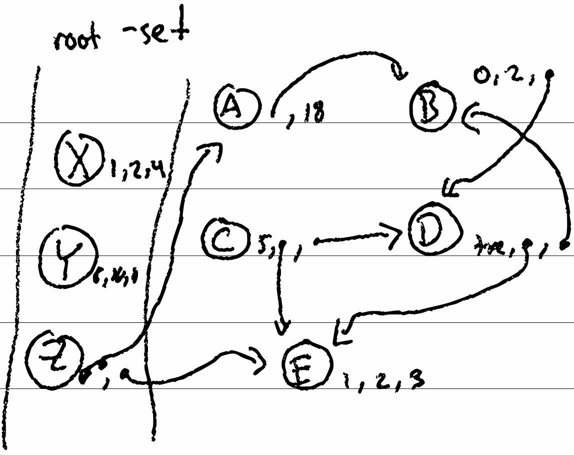
theory: all global variable, ptrs in the prog src, local (stack + registers)

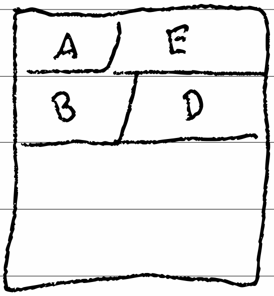
us: in the root stack rs



13-5)







vec layout:

$\boxed{\text{ptr to ty}}$, val₀ , val₁ , val₂ , ...

type layout:

"Iamavector", howmany , ty₀ , ty₁ , ty₂ , ...

13-6) collect)

make new heap

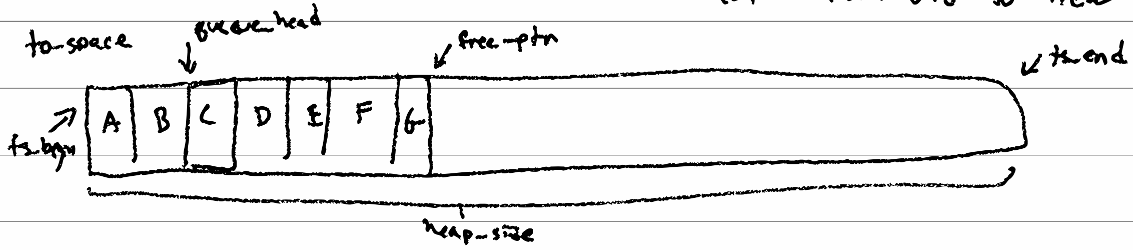
```
scan root set
while q ain't empty
    scan q members
free the old space
```

scan)

walks an object enqueing the things it points to

enqueue)

copies from old to new

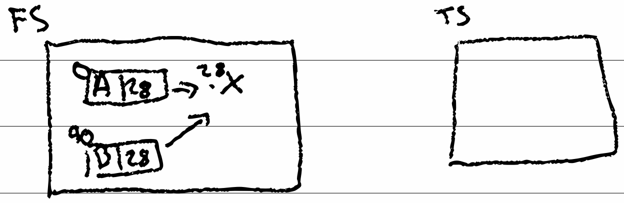


13-7 forwarding pointer

before: obj^X @ addr 28

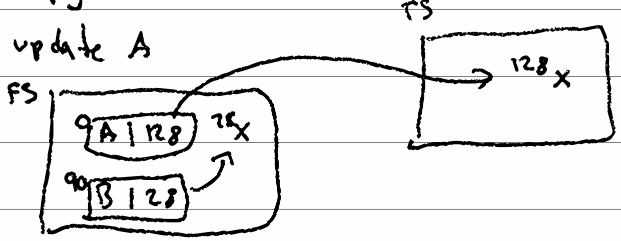
Step 0: scan A — pointed by obj^A @ addr 0

↳ enqueue(28) — + obj^B @ addr 90

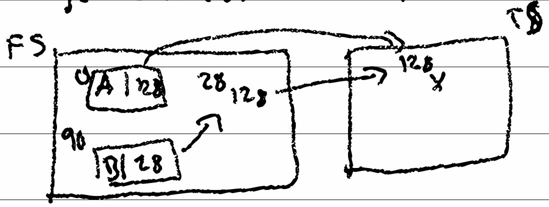


step 1: copy X

step 2: update A

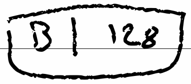


step 3: change old X (28) to a fwd ptr



step 4: scan B ⤳ enqueue 28

step 5: update B



13-8/

```
void scan ( int64_t * obj;) {
    ty = obj [0]
    for ( int i = 0 ; i < ty [1] ; i++ ) {
        elem_ty = ty [2 + i]
        if (elem_ty is ty_vector) {
            enqueue ( & obj [1+i]) } }
    queue_head += ty [1] + 1 }
```

```
13-9) enqueue ( int64_t ** objref_ptr ) {

  new_loc = free_ptr;
  obj = * objref_ptr;
  ty = obj [0]
  if ( ty ∈ To Space ( ts_begin ≤ ty < ts_end)) {
      // forwarding!
       size = 0 ; new_loc = ty }
  else {  size = ty [1] + 1
           for ( .... )  { newloc [i] = obj [i] } }
  free_ptr += size
  * objref_ptr = new_loc
  * obj = new_loc
```