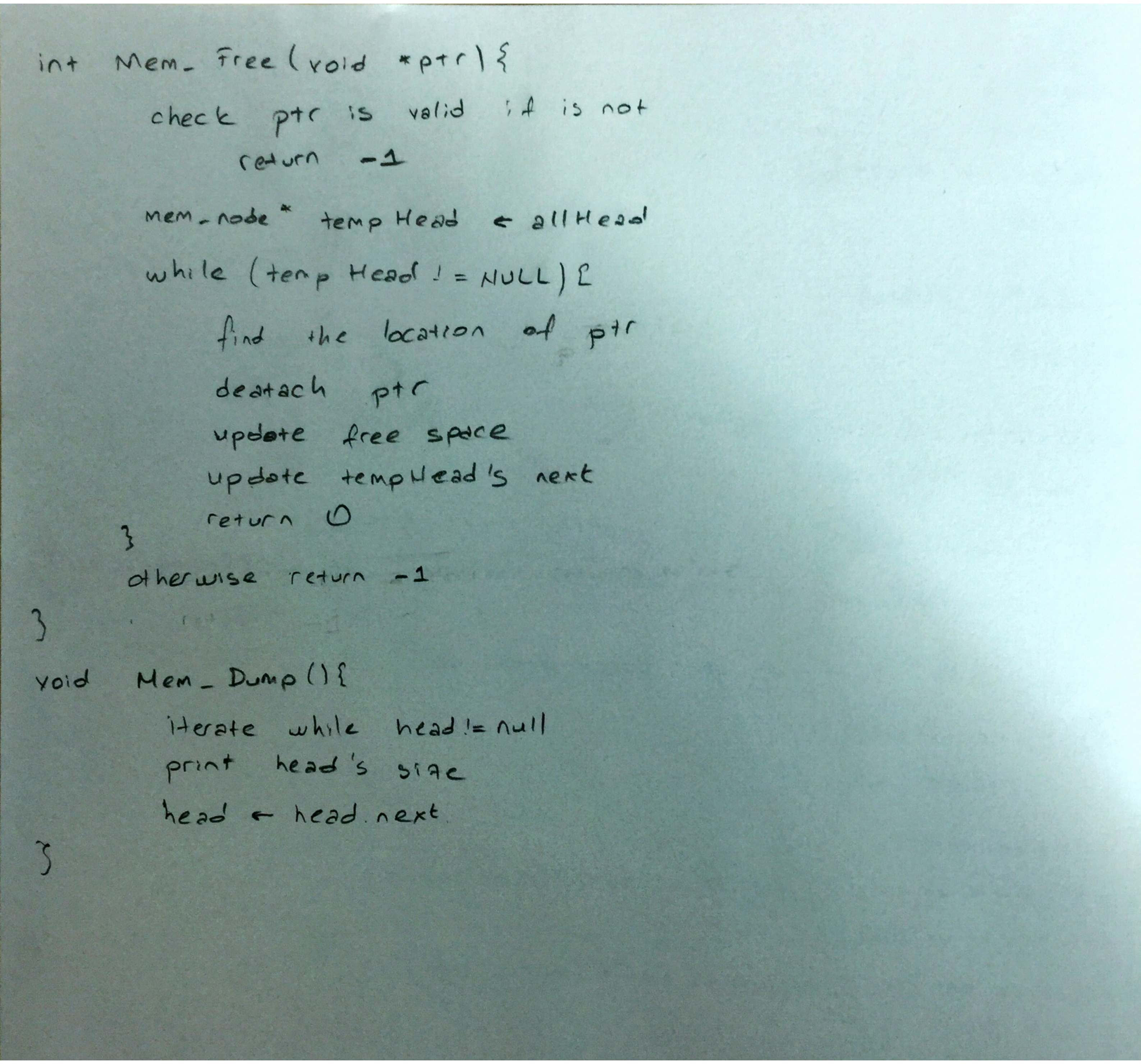
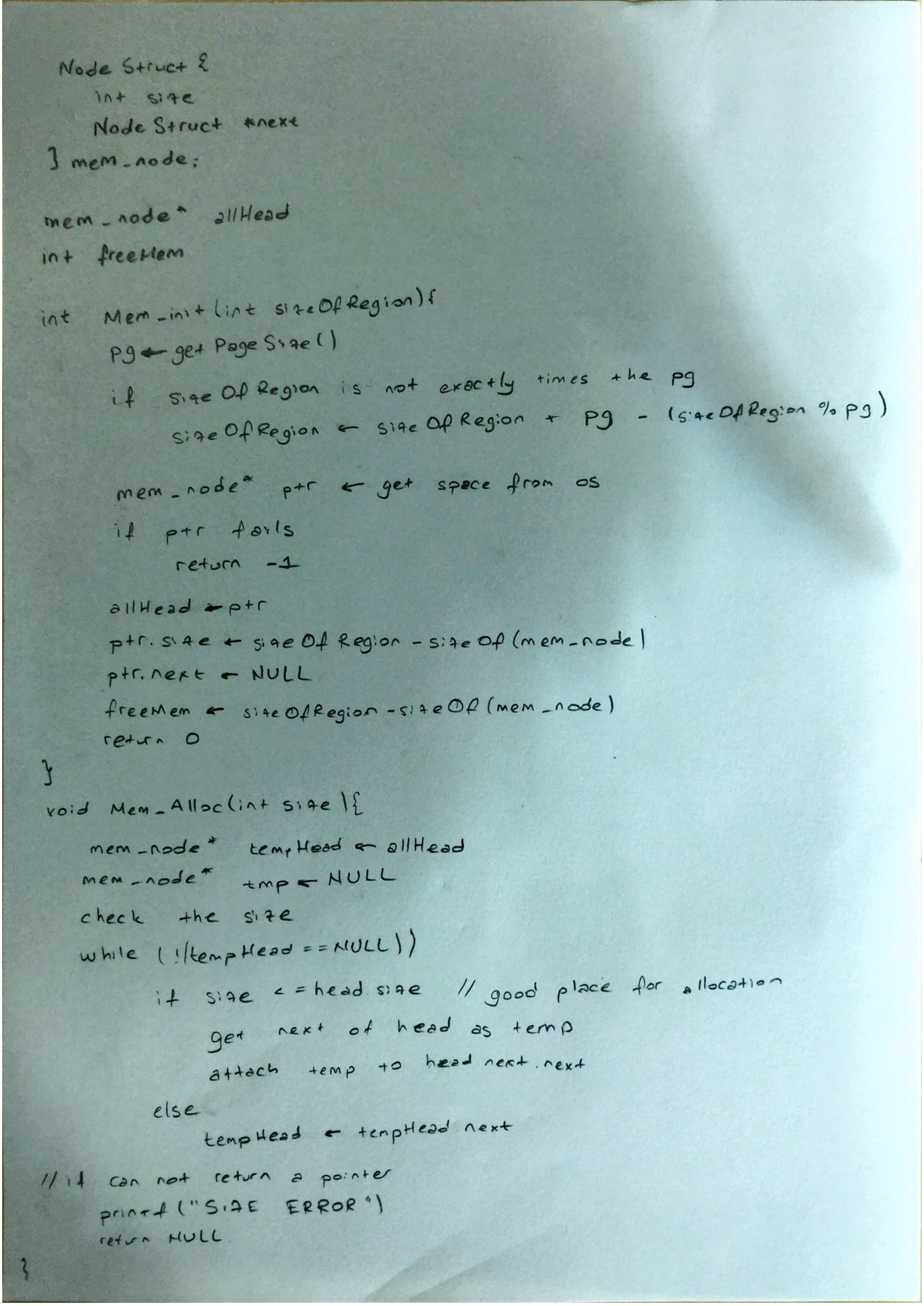


```
int Mem_Free(void *ptr){
      check ptr is valid if is not
            return -1
      mem_node* tempHead ← allHead
      while (tempHead != NULL){
            find the location of ptr
            deatach ptr
            update free space
            update tempHead's next
            return 0
      }
      otherwise return -1
}
void Mem_Dump(){
      iterate while head != null
      print head's size
      head ← head.next
}
```



```
Node Struct {
    int size
    Node Struct *next
} mem_node;

mem_node* allHead
int freeMem

int  Mem_init(int sizeOfRegion){
    pg ← get PageSize()
    if  sizeOfRegion is not exactly times the pg
        sizeOfRegion ← size Of Region + pg - (sizeOfRegion % pg)

    mem_node*  ptr ← get space from os
    if ptr fails
        return -1
    allHead ← ptr
    ptr.size ← sizeOfRegion - sizeOf(mem_node)
    ptr.next ← NULL
    freeMem ← sizeOfRegion - sizeOf(mem_node)
    return 0
}
void Mem_Alloc(int size){
   mem_node*  tempHead ← allHead
   mem_node*  tmp ← NULL
   check the size
   while (!(tempHead == NULL))
       if size <= head.size   // good place for allocation
           get next of head as temp
           attach temp to head next.next
       else
           tempHead ← tempHead.next
// if can not return a pointer
    printf("SIZE ERROR")
    return NULL
}
```