CS 1550

Project 3

Project 3 - Virtual Memory Simulator

- Simulate memory page allocation and page eviction algorithm

Project 3 - Virtual Memory Simulator

- Simulate memory page allocation and page eviction algorithm
  - Your program will read from a memory trace

Project 3 - Virtual Memory Simulator

- Simulate memory page allocation and page eviction algorithm
  - Your program will read from a memory trace

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Simulate memory page allocation and page eviction algorithm
  - Your program will read from a memory trace
  - You will implement how loaded pages are evicted

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Since it is a 32-bit address space.

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

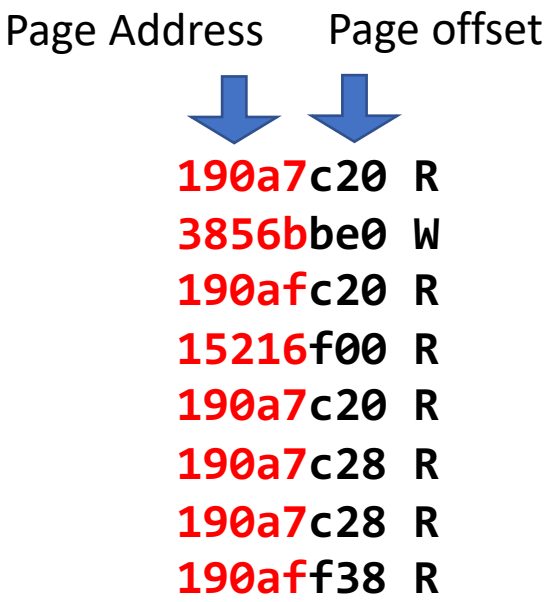
- Since it is a 32-bit address space.
  - First 20 bits is used for the address

Page Address

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Since it is a 32-bit address space.
  - First 20 bits is used for the address
  - The rest is used for offset



Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	
1	
2	

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	
1	
2	

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	
1	
2	

Pagefault since it is not in the process table

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

Pagefault since it is not in the process table

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

0	**190a7**
1	
2	

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	
2	

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	
2	

Pagefault since it is not in the process table

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	**3856b**
2	

Pagefault since it is not in the process table

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	**3856b**
2	

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	**3856b**
2	**190af**

Pagefault since it is not in the process table

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

0	**190a7**
1	**3856b**
2	**190af**

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume FIFO

Pagefault again

0	**190a7**
1	**3856b**
2	**190af**

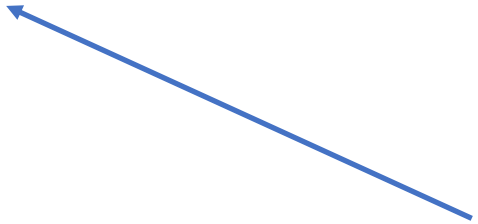
We need to evict someone!!

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume **FIFO**

Pagefault again



We need to evict someone!!

0	**190a7**
1	**3856b**
2	**190af**

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume **FIFO**

Pagefault again

0	**190a7**
1	**3856b**
2	**190af**

We need to evict someone!!

```
190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R
```

Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume **FIFO**

Pagefault again

We need to evict someone!!

190a7c20 R
3856bbe0 W
190afc20 R
15216f00 R
190a7c20 R
190a7c28 R
190a7c28 R
190aff38 R

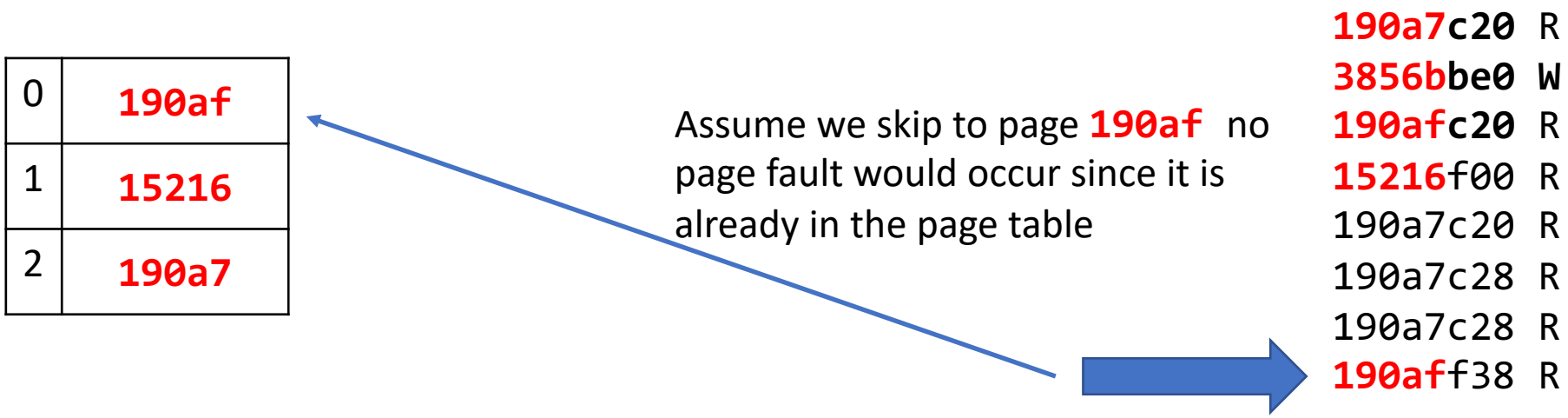
Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - Assume **FIFO**

Pagefault again

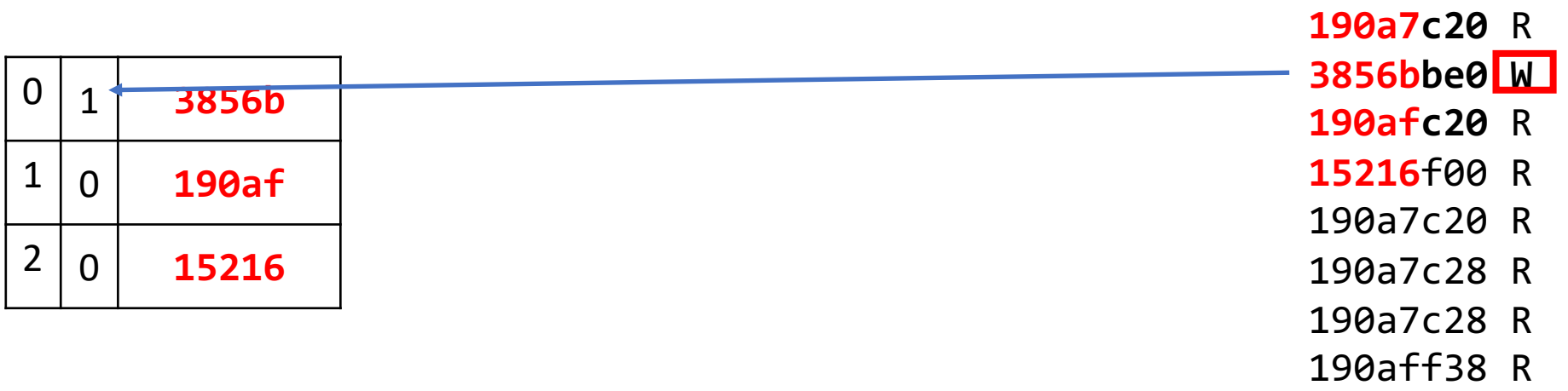
Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB



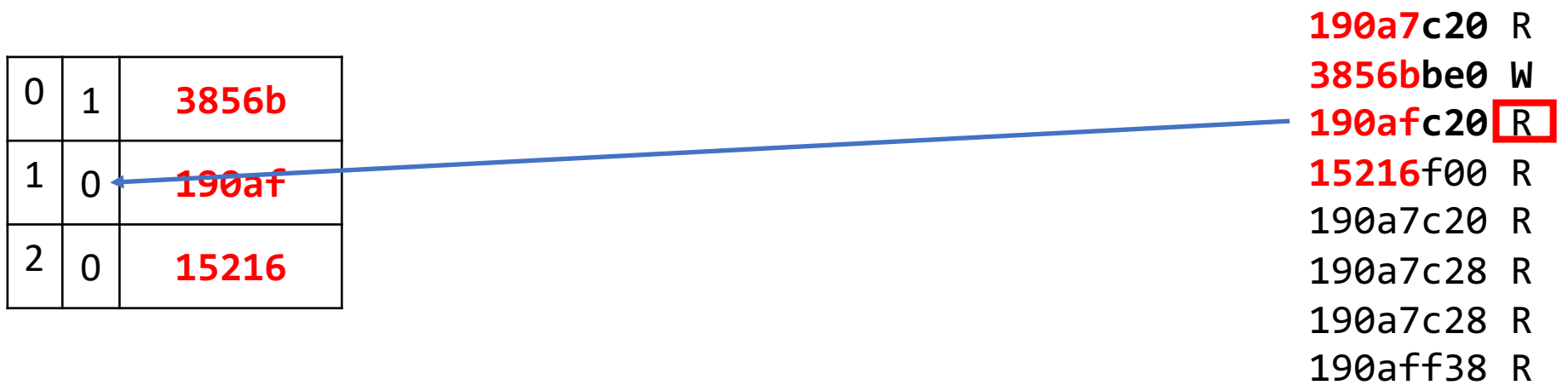
Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - For other algorithms such as not recently used **(NRU)**



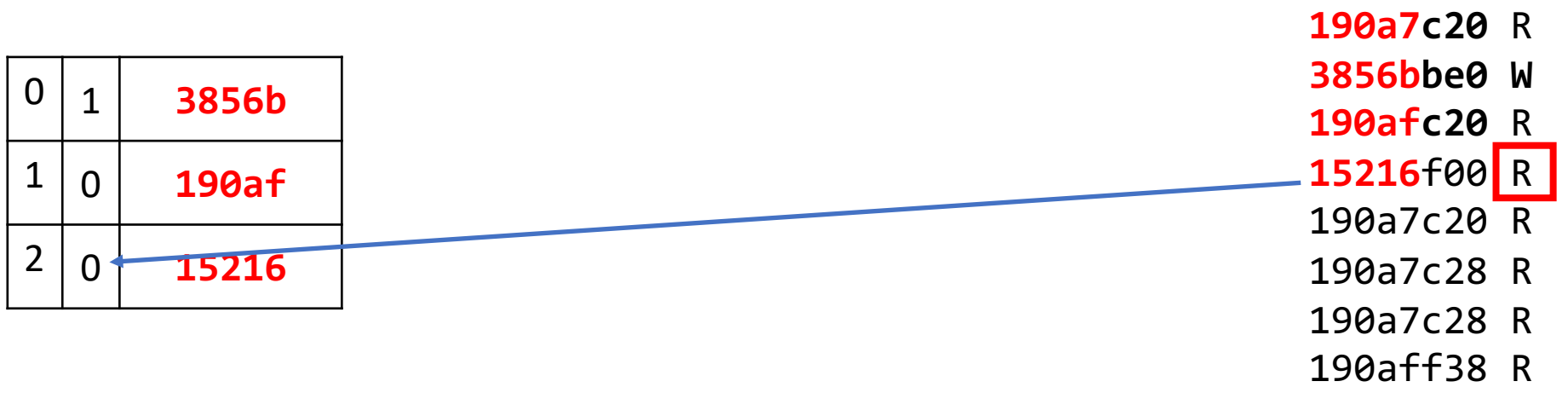
Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - For other algorithms such as not recently used **(NRU)**



Project 3 - Virtual Memory Simulator

- Lets suppose you have 12KB of physical memory
  - Page has 4KB
  - For other algorithms such as not recently used **(NRU)**



Project 3 - Virtual Memory Simulator

- No need to use qemu
- You will write the simulator from scratch with Java, c++,Perl, or Python
- Read from memory traces text files
- Count the number of events (pagefaults, page evictions etc.)
  - Compare eviction algorithms

Sbrk on XV6

- Final tips for lab 3

Sbrk on XV6

- In **sysproc.c** file

```
int
sys_sbrk(void)
{
  int addr;
  int n;

  if(argint(0, &n) < 0)
    return -1;
  addr = myproc()->sz;

  if(growproc(n) < 0)
     return -1;

  return addr;
}
```

Sbrk on XV6

- In **sysproc.c** file

```
int
sys_sbrk(void)
{
  int addr;
  int n;

  if(argint(0, &n) < 0)
    return -1;
  addr = myproc()->sz;

  /*if(growproc(n) < 0)
      return -1; */
 //increament the addr
  return addr; //return the old address
}
```

Sbrk on XV6

```
int
growproc(int n)
{
  uint sz;
  struct proc *curproc = myproc();

  sz = curproc->sz;
  if(n > 0){
    if((sz = allocuvm(curproc->pgdir, sz, sz + n)) == 0)
      return -1;
  } else if(n < 0){
    if((sz = deallocuvm(curproc->pgdir, sz, sz + n)) == 0)
      return -1;
  }
  curproc->sz = sz;
  switchuvm(curproc);
  return 0;
}
```

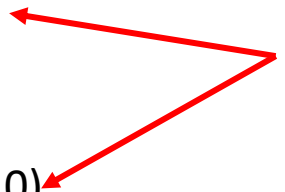
- In **proc.c** file

Sbrk on XV6

- In **proc.c** file

```
int
growproc(int n)
{
  uint sz;
  struct proc *curproc = myproc();

  sz = curproc->sz;
  if(n > 0){
    if((sz = allocuvm(curproc->pgdir, sz, sz + n)) == 0)
      return -1;
  } else if(n < 0){
    if((sz = deallocuvm(curproc->pgdir, sz, sz + n)) == 0)
      return -1;
  }
  curproc->sz = sz;
  switchuvm(curproc);
  return 0;
}
```



Allocation and deallocation

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
   a = PGROUNDUP(oldsz);
   mem = kalloc();
   for(; a < newsz; a += PGSIZE){
         memset(mem, 0, PGSIZE);

         if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
         …
   }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
        allocuvm(curproc->pgdir, sz, sz + n)
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
   a = PGROUNDUP(oldsz);
   mem = kalloc();
   for(; a < newsz; a += PGSIZE){
        memset(mem, 0, PGSIZE);

        if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
        …
   }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
        allocuvm(curproc->pgdir, sz, sz + n)
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
   a = PGROUNDUP(oldsz);
   mem = kalloc();
   for(; a < newsz; a += PGSIZE){
        memset(mem, 0, PGSIZE);

        if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
        …
   }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
        allocuvm(curproc->pgdir, sz, sz + n)
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
   a = PGROUNDUP(oldsz);
   mem = kalloc();
   for(; a < newsz; a += PGSIZE){
        memset(mem, 0, PGSIZE);

        if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
        …
   }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
    a = PGROUNDUP(oldsz);
    mem = kalloc();
    for(; a < newsz; a += PGSIZE){
            memset(mem, 0, PGSIZE);

            if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
            …
    }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

Allocates 4096 B of physical memory and returns a virtual address

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

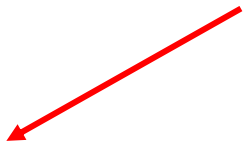
Ensure memory is clean

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```



Maps address to process table virtual address

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

Process Page table

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```

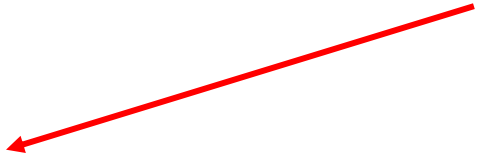
Translating virtual address to physical address

Sbrk on XV6

- In file **vm.c**

```
int
allocuvm (pde_t *pgdir, uint oldsz, uint newsz)
{
…
    mem = kalloc();
…
    memset(mem, 0, PGSIZE);

    if(mappages(pgdir, (char*)a, PGSIZE, V2P(mem), PTE_W|PTE_U) < 0){
…
    }
  }
  return newsz;
}
```



Flags the page as writeable and to be used by programs (otherwise only the kernel can access it).

Sbrk on XV6

- In file **trap.c** and method **trap()**

```
void
trap(struct trapframe *tf)
…
  //cases
…
  //PAGEBREAK: 13
  default:
….
    // In user space, assume process misbehaved.
    cprintf("pid %d %s: trap %d err %d on cpu %d "
            "eip 0x%x addr 0x%x--kill proc\n", myproc()->pid, myproc()->name, tf->trapno, tf->err, cpuid(), tf->eip, rcr2());

    myproc()->killed = 1;
```

Sbrk on XV6

- In file **trap.c** and method **trap()**

```
void
trap(struct trapframe *tf)
…
case T_PGFLT:
   growproc(4092);
   break;

  //PAGEBREAK: 13
  default:
….
    // In user space, assume process misbehaved.
    cprintf("pid %d %s: trap %d err %d on cpu %d "
            "eip 0x%x addr 0x%x--kill proc\n", myproc()->pid, myproc()->name, tf->trapno, tf->err, cpuid(), tf->eip, rcr2());

    myproc()->killed = 1;
```

Sbrk on XV6

- In file **trap.c** and method **trap()**

```
void
trap(struct trapframe *tf)
…
case T_PGFLT:
  // add just one frame. Exactly where the user requests adjusted by page start address.
  break;

 //PAGEBREAK: 13
 default:
….
   // In user space, assume process misbehaved.
   cprintf("pid %d %s: trap %d err %d on cpu %d "
           "eip 0x%x addr 0x%x--kill proc\n", myproc()->pid, myproc()->name, tf->trapno, tf->err, cpuid(), tf->eip, rcr2());

   myproc()->killed = 1;
```

Sbrk on XV6

- In file **trap.c** and method **trap()**

```
void
trap(struct trapframe *tf)
…
case T_PGFLT:
  // add just one frame. Exactly where the user requests adjusted by page start address.
  PGROUNDDOWN(rcr2());
  // alloc memory, clean and map it in the process page table.
  break;

 //PAGEBREAK: 13
 default:
….
   // In user space, assume process misbehaved.
   cprintf("pid %d %s: trap %d err %d on cpu %d "
           "eip 0x%x addr 0x%x--kill proc\n", myproc()->pid, myproc()->name, tf->trapno, tf->err, cpuid(), tf->eip, rcr2());

   myproc()->killed = 1;
```