OS-II: CS3523
Programming Assignment 5: xv6 Paging

Understanding System Calls:

Think of system calls as a way for regular programs to ask the operating system (OS), the big boss of the computer, to do something special for them.
These special tasks can range from reading files to creating new processes, and they're all managed by the OS through system calls.

When a program needs the OS to do something for it, it sends a message called a system call and waits for the OS to handle the request.
Key Learnings from the Assignment:

Creating Custom Requests:
The assignment introduced me to the concept of creating custom system calls within the xv6 operating system.
This involved defining specific tasks I wanted the OS to perform, such as printing memory information or monitoring memory usage.
Understanding Memory Management:
Through the assignment, I gained insights into memory management and how the OS organizes memory using page tables.
I learned how to examine these page tables to gain a better understanding of memory utilization and allocation.

Communication with the OS:
I learned about the proper protocols for communicating with the OS, ensuring that requests and information are transmitted correctly.
Effective communication is crucial to ensure that the OS understands the tasks requested and executes them accurately.
Error Handling and Security Measures:
The assignment emphasized the importance of error handling and implementing security measures.
I learned techniques for validating user input, detecting errors, and ensuring the security of system operations to prevent unauthorized access.
Conducting Experimental Analysis:

The assignment involved conducting experiments to observe the behavior of system calls under various conditions.
Through these experiments, I gained practical insights into the performance and behavior of the OS in response to different tasks and scenarios.

Task-1: Printing the page table entries

Output 1:- No array declaration

```
xv6 kernel is booting

hart 2 starting
hart 1 starting
init: starting sh
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f40000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f3d000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f3b000
$
```

Output 2:-Global array declaration

```
xv6 kernel is booting

hart 1 starting
hart 2 starting
init: starting sh
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f40000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f3d000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page address: 0x0000000087f3c000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f3b000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address: 0x0000000087f3a000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page address: 0x0000000087f39000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page address: 0x0000000087f38000
PTE No: 7, Virtual page address: 0x0000000000007000, Physical page address: 0x0000000087f37000
PTE No: 8, Virtual page address: 0x0000000000008000, Physical page address: 0x0000000087f36000
PTE No: 9, Virtual page address: 0x0000000000009000, Physical page address: 0x0000000087f35000
PTE No: 10, Virtual page address: 0x000000000000a000, Physical page address: 0x0000000087f34000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page address: 0x0000000087f32000
```

Output 3:-Local array declaration

```
xv6 kernel is booting

hart 1 starting
hart 2 starting
init: starting sh
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f40000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f3d000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f3b000
```

Output 4:-Running multiple Times

```
xv6 kernel is booting

hart 2 starting
hart 1 starting
init: starting sh
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f40000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f3d000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f3b000
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f54000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f63000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f6e000
$ mypgtPrint
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address: 0x0000000087f73000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address: 0x0000000087f40000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address: 0x0000000087f3e000
$
```

Observations:-

The increase in the number of page table entries with user access is expected due to the user's access to the global array, necessitating the user bit to be set to 1.

The number of valid page table entries increases from 3 to 12 because the global array requires 40,000 bytes of storage, spanning 9 pages, each accommodating 1024 integers.

The stack of a process remains a single page, as allocated by the exec system call, leading to no change in the number of valid entries after declaring a local array.

Compiler optimization ensures efficient stack usage, typically requiring only a single page for program execution, hence no increase in the number of page table entries.

Repeating the execution of the user program doesn't change the number of entries in the page table, as each execution creates a new process with its own virtual address space mapping to physical memory.

Virtual addresses remain consistent across multiple executions, while physical addresses differ due to the allocation of new physical memory for each process.

Solution Methodology For Task 1:-

Defined the mypgtPrint() Function:

Obtained the current process using myproc().

Accessed the root of the multi-level page table from the process structure. And then Called the printPTEs() function to start printing from the root.

Implemented the sys_mypgtPrint() System Call:

Defind the system call sys_mypgtPrint() to invoke the mypgtPrint() function.

Implemented the printPTEs() Function:

Defined the printPTEs() function to traverse and print the page table entries recursively.
Checked if the current entry is valid and user accessible (PTE_V and PTE_U flags).
If at the leaf level, print the PTE number, virtual page address, and physical page address.
If not at the leaf level, recursively traverse to the next level of the page table.
Invoked the System Call:
In the user program, invoke the mypgtPrint() system call.

Task-2: Implement demand paging

Output 1:- When a global array is decalred

```
xv6 kernel is booting

hart 2 starting
hart 1 starting
init: starting sh
$ mypgdemand
Page fault has occured, doing demand paging for: 0x0000000000005008
Page mapped successfully at address: 0x0000000000005000
Page fault has occured, doing demand paging for: 0x0000000000014f48
Page mapped successfully at address: 0x0000000000014000
Page fault has occured, doing demand paging for: 0x0000000000001010
Page mapped successfully at address: 0x0000000000001000
global addr from user space: 1010
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000002000
Page mapped successfully at address: 0x0000000000002000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000003000
Page mapped successfully at address: 0x0000000000003000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000004000
Page mapped successfully at address: 0x0000000000004000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000005000
Page mapped successfully at address: 0x0000000000005000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000006000
Page mapped successfully at address: 0x0000000000006000
```

```
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page Address 0x0000000087f61000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000007000
Page mapped successfully at address: 0x0000000000007000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page Address 0x0000000087f61000
PTE No: 7, Virtual page address: 0x0000000000007000, Physical page Address 0x0000000087f53000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000008000
Page mapped successfully at address: 0x0000000000008000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page Address 0x0000000087f61000
PTE No: 7, Virtual page address: 0x0000000000007000, Physical page Address 0x0000000087f53000
PTE No: 8, Virtual page address: 0x0000000000008000, Physical page Address 0x0000000087f54000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x0000000000009000
Page mapped successfully at address: 0x0000000000009000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page Address 0x0000000087f61000
PTE No: 7, Virtual page address: 0x0000000000007000, Physical page Address 0x0000000087f53000
PTE No: 8, Virtual page address: 0x0000000000008000, Physical page Address 0x0000000087f54000
PTE No: 9, Virtual page address: 0x0000000000009000, Physical page Address 0x0000000087f55000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
Page fault has occured, doing demand paging for: 0x000000000000a000
Page mapped successfully at address: 0x000000000000a000
```

```
Page fault has occured, doing demand paging for: 0x000000000000a000
Page mapped successfully at address: 0x000000000000a000
Printing final page table:
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page Address 0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page Address 0x0000000087f52000
PTE No: 2, Virtual page address: 0x0000000000002000, Physical page Address 0x0000000087f73000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page Address 0x0000000087f72000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page Address 0x0000000087f71000
PTE No: 5, Virtual page address: 0x0000000000005000, Physical page Address 0x0000000087f70000
PTE No: 6, Virtual page address: 0x0000000000006000, Physical page Address 0x0000000087f61000
PTE No: 7, Virtual page address: 0x0000000000007000, Physical page Address 0x0000000087f53000
PTE No: 8, Virtual page address: 0x0000000000008000, Physical page Address 0x0000000087f54000
PTE No: 9, Virtual page address: 0x0000000000009000, Physical page Address 0x0000000087f55000
PTE No: 10, Virtual page address: 0x000000000000a000, Physical page Address 0x0000000087f6f000
PTE No: 12, Virtual page address: 0x000000000000c000, Physical page Address 0x0000000087f4a000
```

Output 2:- When heap is created, “Create a large integer array of size 10000 on heap using malloc() within the main function”

```
$ mypgtPrint
page fault occurred, doing demand paging for address: 0x0000000000005008
page fault occurred, doing demand paging for address: 0x0000000000014f48
page fault occurred, doing demand paging for address: 0x0000000000001000
page fault occurred, doing demand paging for address: 0x0000000000004008
page fault occurred, doing demand paging for address: 0x000000000000f1d8
malloc addr from user space: F1E0
page fault occurred, doing demand paging for address: 0x0000000000010000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address:  0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address:  0x0000000087f52000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address:  0x0000000087f4a000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address:  0x0000000087f73000
PTE No: 15, Virtual page address: 0x000000000000f000, Physical page address:  0x0000000087f72000
PTE No: 16, Virtual page address: 0x0000000000010000, Physical page address:  0x0000000087f71000
page fault occurred, doing demand paging for address: 0x0000000000011000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address:  0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address:  0x0000000087f52000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address:  0x0000000087f4a000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address:  0x0000000087f73000
PTE No: 15, Virtual page address: 0x000000000000f000, Physical page address:  0x0000000087f72000
PTE No: 16, Virtual page address: 0x0000000000010000, Physical page address:  0x0000000087f71000
PTE No: 17, Virtual page address: 0x0000000000011000, Physical page address:  0x0000000087f70000
page fault occurred, doing demand paging for address: 0x0000000000012000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address:  0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address:  0x0000000087f52000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address:  0x0000000087f4a000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address:  0x0000000087f73000
PTE No: 15, Virtual page address: 0x000000000000f000, Physical page address:  0x0000000087f72000
PTE No: 16, Virtual page address: 0x0000000000010000, Physical page address:  0x0000000087f71000
PTE No: 17, Virtual page address: 0x0000000000011000, Physical page address:  0x0000000087f70000
PTE No: 18, Virtual page address: 0x0000000000012000, Physical page address:  0x0000000087f61000
page fault occurred, doing demand paging for address: 0x0000000000013000
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address:  0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address:  0x0000000087f52000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address:  0x0000000087f4a000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address:  0x0000000087f73000
PTE No: 15, Virtual page address: 0x000000000000f000, Physical page address:  0x0000000087f72000
PTE No: 16, Virtual page address: 0x0000000000010000, Physical page address:  0x0000000087f71000
PTE No: 17, Virtual page address: 0x0000000000011000, Physical page address:  0x0000000087f70000
PTE No: 18, Virtual page address: 0x0000000000012000, Physical page address:  0x0000000087f61000
PTE No: 19, Virtual page address: 0x0000000000013000, Physical page address:  0x0000000087f53000
Printing final page table:
PTE No: 0, Virtual page address: 0x0000000000000000, Physical page address:  0x0000000087f4e000
PTE No: 1, Virtual page address: 0x0000000000001000, Physical page address:  0x0000000087f52000
PTE No: 3, Virtual page address: 0x0000000000003000, Physical page address:  0x0000000087f4a000
PTE No: 4, Virtual page address: 0x0000000000004000, Physical page address:  0x0000000087f73000
PTE No: 15, Virtual page address: 0x000000000000f000, Physical page address:  0x0000000087f72000
PTE No: 16, Virtual page address: 0x0000000000010000, Physical page address:  0x0000000087f71000
PTE No: 17, Virtual page address: 0x0000000000011000, Physical page address:  0x0000000087f70000
PTE No: 18, Virtual page address: 0x0000000000012000, Physical page address:  0x0000000087f61000
PTE No: 19, Virtual page address: 0x0000000000013000, Physical page address:  0x0000000087f53000
```

Observations:-

Page faults are occurring at various virtual addresses, such as 0x0000000000005008, 0x0000000000014f48, 0x0000000000001010, etc.

Each time a page fault occurs, demand paging is triggered to resolve it.

The virtual addresses are rounded down to the nearest page boundary before mapping the pages.

For each page fault, physical memory is allocated and initialized with zeros, then mapped to the page table.
The page mapping process seems to be successful each time, as indicated by the message "Page mapped successfully at address: ..."
After multiple page faults, the final page table is printed, showing the mapping between virtual and physical addresses.
The page table entries (PTEs) are numbered sequentially, starting from 0 and incrementing by one for each entry.
Each PTE entry lists the virtual page address and its corresponding physical page address.

Solution Methodology:-

Identified Trigger Conditions:
Understood the conditions that trigger page faults in user space. In this case, page faults occur when r_scause() is 13 or 15.
Checked Virtual Address:
Extracted the virtual address that caused the page fault. This information is typically available as an argument to printf() in usertrap().
Rounded Address:
Rounded down the faulting virtual address to a page boundary using PGROUNDDOWN(va).
Allocated Physical Memory:
Allocated a new page of physical memory to map at the faulting address. This can be done using a memory allocation function like kalloc().
Mapped Page to Page Table:
Mapped the newly allocated physical page to the faulting virtual address in the process's page table. Ensure appropriate permissions (PTE_W|PTE_R|PTE_X|PTE_U) are set.
Handled Errors:
Handled any errors that may occur during the allocation or mapping process. If an error occurs, mark the process as killed (p->killed = 1) to terminate it.
Completed Handling:
After mapping the page, returned control back to the user space process to let it continue executing.

Task-3: Implement logic to detect which pages have been accessed and/or dirty

Output:-

Observations:-

Page Access Information: The output shows information about the access and dirty status of each page. Each page is listed with its page number, accessed status, and dirty status.

Accessed Status: Indicates whether the page has been accessed since the last check. A value of 1 means the page has been accessed, while 0 means it hasn't.
Dirty Status: Indicates whether the page has been modified (dirty). A value of 1 means the page has been modified, while 0 means it hasn't.
Page Numbers and Status: Each page from 0 to 19 is listed along with its accessed and dirty status.
For example, page 0 and page 1 have both been accessed but are not dirty (modified).
Pages from 2 to 19 have not been accessed or modified.
System Call Invocation: The system call mypgaccess is invoked with arguments 0 20 0.
The starting virtual address of the first user page to check is 0.
The number of pages to check is 20.
The user address to a buffer to store the results is 0.

Solution Methodology for Task 3:-

Defined the mypgaccess() Function:
Defined a function named mypgaccess() that takes no arguments and returns an integer.
Inside the function, declare variables addrbegin, numpge, and useradr to store the address range, number of pages, and user address provided as arguments.
Used argaddr() and argint() to retrieve the arguments passed from user space.
Initialized bitmask to zero to track accessed and dirty pages.
Created a complement bitmask complaccess and compldirty to clear the accessed and dirty bits.
Obtained the current process using myproc().
Iterated through each page in the specified address range.
Used walk() function to get the page table entry (pte) corresponding to the virtual address.
Checedk if the accessed (PTE_A) or dirty (PTE_D) bit is set in the page table entry.
If the accessed bit is set, updated the bitmask and clear the accessed bit in the page table entry.
If the dirty bit is set, updated the bitmask and clear the dirty bit in the page table entry.
Used copyout() function to copy the bitmask to the user address space.
Returned 0 to indicate successful completion of the function.
Defined the System Call:
Defined a system call named sys_mypgaccess() to invoke the mypgaccess() function.
Invoke the System Call:
In the user program, invokde the sys_mypgaccess() system call by making a system call with appropriate arguments.