Operating Systems

Assignment – 9

1. IMPLEMENTATION OF MEMORY MANAGEMENT USING PAGING
    a. Write a C program to simulate the Paging technique of memory management.

        - Following steps should be implemented in the program:

            - Ask the user to enter memory size.Ask the user to enter page size
            - Display number of pages in memory according to the acquired data
            - Ask the user to input the number of processes and pages required for each process
            - Enter the page table of each process
            - If enough memory is available, assign memory to each process otherwise display the message “Memory is Full”
            - Compute and display the physical address of a memory address by asking the user to enter process number, page number, and offset.
            - If the user inputs correct data, compute and display the corresponding physical address, otherwise display an error message

Code:

```c
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

int main()
{
    int mem_size, page_size, num_pages, num_processes, i, j, k, process_num, page_num, offset, physical_addr, flag = 0;
    printf("Enter the memory size: ");
    scanf("%d", &mem_size);
    printf("Enter the page size: ");
    scanf("%d", &page_size);
    num_pages = mem_size/page_size;
    printf("Number of pages in memory: %d\n", num_pages);
    printf("Enter the number of processes: ");
    scanf("%d", &num_processes);
    int *page_table[num_processes];
    for(i = 0; i < num_processes; i++)
    {
        printf("Enter the number of pages required for process %d: ", i+1);
        int num_pages_proc;
        scanf("%d", &num_pages_proc);
        if(num_pages_proc > num_pages)
        {
            printf("Memory is Full\n");
            exit(0);
        }
        else
        {
            page_table[i] = (int*)malloc(num_pages_proc*sizeof(int));
            printf("Enter the page table for process %d: ", i+1);
            for(j = 0; j < num_pages_proc; j++)
            {
                scanf("%d", &page_table[i][j]);
                if(page_table[i][j] >= num_pages)
                {
                    printf("Invalid page number %d\n", page_table[i][j]);
                    flag = 1;
                    break;
                }
            }
```

```
            if(flag == 1)
            {
                break;
            }
        }
    }
    if(flag == 1)
    {
        exit(0);
    }
    printf("Enter the process number: ");
    scanf("%d", &process_num);
    printf("Enter the page number: ");
    scanf("%d", &page_num);
    printf("Enter the offset: ");
    scanf("%d", &offset);
    if(process_num > num_processes || page_num >= page_table[process_num-1][j] || offset >= page_size)
    {
        printf("Invalid input\n");
        exit(0);
    }
    physical_addr = (page_table[process_num-1][page_num] * page_size) + offset;
    printf("The physical address is %d\n", physical_addr);
    return 0;
}
```

Output:

```
Enter the memory size: 4096
Enter the page size: 512
Number of pages in memory: 8
Enter the number of processes: 2
Enter the number of pages required for process 1: 3
Enter the page table for process 1: 0 1 3
Enter the number of pages required for process 2: 2
Enter the page table for process 2: 2 4
Enter the process number: 1
Enter the page number: 2
Enter the offset: 256
The physical address is 1792


...Program finished with exit code 0
```


Presented By:

Harshita Pasupuleti

21BCE8421