Cameron Hartling
ID: 007735059

CSE 4600 : Operating Systems
Homework 1

Question 1. (2 points) Please describe the essential components of a processor (CPU) and elucidate the process by which a processor executes machine language instructions.

The CPU consists of an arithmetic logic unit (ALU), a control unit, and processor registers. The ALU performs bitwise and arithmetic operations. The ALU receives operands and code to indicate the operation to be performed and returns the result. The processor registers reads, writes, and stores data that the CPU is using. The control unit directs the operation of the CPU. The control unit will send operands and directions to the ALU and redirect the results to the processor registers or a more permanent storage located elsewhere in the computer. It uses a binary decoder to interpret the machine language instructions to regulate the timing of control signals of the other units of a computer.

Question 2. (2 points) Please provide a detailed explanation of the two fundamental functions of an operating system (OS).

The OS provides a clean, easy-to-use interface for the user and provides resource management. For the user, the OS provides interfaces to hardware resources, error detection, input/output operations, and program execution. It also manages cache, memory, mass-storage, file-system, and the I/O system. Lastly, the OS provides protection and security.

Question 3. (2 points) Please discuss the concept of system calls and the role of the standard C Library (glibc) in the Linux operating system.

System calls are used to provide an interface for the services of an operating system. The role of glibc is to allow the programmer access to the API in order to run system calls.

Question 4. (2 points) Please explain the concept of user mode and kernel mode in an operating system. Describe the key differences between these two modes, and discuss why the distinction between user mode and kernel mode is important for the security and stability of the operating system.

The dual mode operation allows for the OS to protect itself and the system components from errors made by users. The kernel mode has full access to all capability of the hardware, area of the memory, and the full instruction set. The user mode has no access to the hardware directly and only a partial instruction set and limited memory access. The user mode has to make a valid request to switch into kernel mode and run system calls. The distinction of the two modes allows for a security system to be put into place to make sure the user is not intending harm to the system components or the OS.

Cameron Hartling
ID: 007735059

Question 5. (2 points) Please provide comprehensive explanations of the concepts of multiprogramming and multitasking, with a particular emphasis on elucidating how multiprogramming effectively enhances CPU utilization.

In multiprogramming there are several processes in memory and when one process has finished executing or is waiting for I/O the next process in the memory starts up. Multiprocessing focuses on making it so that the CPU always has a process to run. Multitasking works so that each process has a set time to run and then it swaps to the next process. This allows the user to access all of the processes at once.

Question 6. (20 points)

- **Prompt the user to enter a backup directory name.**
- **If the backup directory does not exist in the current directory, create the backup directory and print a message.**
- **For each .cpp file in the current directory,**
    - **If there have been changes to the file since the last backup or no copy exists in the backup directory, copy the current .cpp file to the backup directory and print a message indicating that the file has been backed up.**
    - **Otherwise, no copy will be made, and print a message stating that the file is the latest.**

Screenshots

- **When no copies of .cpp files exist in the backup directory, your program should back up all .cpp files to the backup directory.**

```
[007735059@csusb.edu@jb358-5 hw1]$ ./hw1.sh
Enter the directory you wish to backup:
backup
Made backup directory.
File has been successfully backed up.
File has been successfully backed up.
File has been successfully backed up.
```

- **When all .cpp files in the backup directory are the latest, your program should not make any copies to the backup directory.**

```
[007735059@csusb.edu@jb358-5 hw1]$ ./hw1.sh
Enter the directory you wish to backup:
backup
Found backup directory.
File in the backup directory is up to date
File in the backup directory is up to date
File in the backup directory is up to date
```

Cameron Hartling
ID: 007735059

- **When there have been changes to a .cpp file since the last backup, your program should back up or copy the updated file to the backup directory.**

```
[007735059@csusb.edu@jb358-5 hw1]$ vi main.cpp
[007735059@csusb.edu@jb358-5 hw1]$ ./hw1.sh
Enter the directory you wish to backup:
backup
Found backup directory.
File in the backup directory is up to date
File in the backup directory is up to date
File has been successfully backed up.
File in the backup directory is up to date
```

- **When a new .cpp file is added to the current directory, your program should be able to back up or copy the new file to the backup directory.**

```
[007735059@csusb.edu@jb358-5 hw1]$ ./hw1.sh
Enter the directory you wish to backup:
backup
Made backup directory.
File has been successfully backed up.
File has been successfully backed up.
File has been successfully backed up.
[007735059@csusb.edu@jb358-5 hw1]$ touch addition.cpp
[007735059@csusb.edu@jb358-5 hw1]$ ls
addition.cpp  backup  function.cpp  hw1.sh  main.cpp  test.cpp
[007735059@csusb.edu@jb358-5 hw1]$ ./hw1.sh
Enter the directory you wish to backup:
backup
Found backup directory.
File has been successfully backed up.
File in the backup directory is up to date
File in the backup directory is up to date
File in the backup directory is up to date
[007735059@csusb.edu@jb358-5 hw1]$
```