HomeWork 2, OS Principles

Due Date: Wednesday 10/13/2021
HW Delivery: submit on **Canvas** by the due date, before midnight
Total Points: 60

General Rules: Create homework, compose specifications or any text by using a common *document-creation* tool, such as Microsoft® Word. Each question is worth 5 points.
To complete your work, consult lecture notes and the www.

1. Characterize high-level the original ***Unix OS*** design. State key novel ideas (at the time) and SW design goals. List design advantages. How relevant is Unix today? Who were original developers? Initial development motivation?

    - Unix operating system was developed by Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy, and Joe Ossanna at Massachusetts Institute of Technology (MIT), AT&T Bell Labs in mid-1960.
    - Unix works as the direct link between the computer system and a user or set of users.
    - Unix was the first os system created in a C Language that made it very easy to transfer between different computers with little modifications.
    - Unix has a hierarchical file system that makes data access and management easier.
    - Unix includes networking capabilities that allow various users to effortlessly share intelligence and information.
    - The UNIX operating system is a multi-user OS that is extensively used on Web-server, computer systems, and workstations.
    - As Unix OS is open-source so several systems are modified or written over the original Unix code, hence no original Unix code remains
    - The primary characteristics of UNIX operating systems are listed below.
        - Unix is transportable (the ability to run on many different systems).
        - Unix has multitasking and multiuser features.
        - Unix has a large software library, making it relevant and valuable when it was created in 1969.
    - At present time Unix is the most powerful and famous operating system.
    - For strict security, authentication features, and a multi-purpose use-case, Unix operating system is most useful.
    - Unix allows the user to perform protective memory-operations.
    - Unix allows the users to execute multiple programs at the same time. So it is multi-tasking operating system.

2. Name and briefly outline some ideal, high-level OS design goals.

- A high-level design gives you a bird's-eye view of a system, product, service, or process. The highest-level design should contain a concise description of the platforms, systems, goods, services, and processes on which it relies, as well as any significant modifications that must be made to them. A high-level design is a representation of a system, product, service, or process at a high level.
  This kind of perspective makes it easier for supporting components to work together.
  The highest-level design should contain a concise description of the platforms, systems, goods, services, and processes on which it relies, as well as any significant modifications that must be made to them. The goal is to briefly highlight each work area, explicitly assigning responsibility of more comprehensive design work while also promoting efficient communication amongst project teams. In the high-level design, each type of end-user should be recognized, and each contributing design should take customer experience into account.

3. Write a concise English language description of the C/C++ function **fflush()**.
   - fflush() is a part of the C/C++ library and it is used for streaming of the output that must be given in the code. The main purpose is used toi buffer the output and forward the data to console output. Syntax: fflush(int *stream)

4. Compose a concise description of the Unix/Linux OS command **whoami**. Also try WHOAMI on MS Windows; describe and document what you observe.
   - When you have different usernames on several frequently used UNIX computers. You may invoke whoami to determine your current username and, hence, which computer you are using. You might also use whoami when sharing a terminal with other users. If the shared terminal has been left unattended, you can issue whoami to list the name of the user who is currently logged in. You may then locate that user and to end the terminal session. The function performed by this is the same as the command id -un in the terminal. In Open VMS, the corresponding command SHOW PROCESS returns the current user's username, as well as other useful information; whoami returns only the username.

5. Write C++ program ***system1*** that calls library function "**system()**". Issue 3 distinct calls to system(), each with a single argument: "ls", "pwd", and "whoami". Show the source and generated outputs. Describe briefly what you observe.

Plausible run-time output:

```
herberts-Air-2:~ herbertmayer$ ./a.out
argc = 1
next call: 'system(ls)'.
Applications    Downloads   Music       c.c             herb_12_16
Contacts        Favorites   Pictures    familie         pix_2020_report
Desktop         Library     Public      familie_12_16   pix_low_2020
Documents       Movies      a.out       herb

next call: 'system(pwd)'.
/Users/herbertmayer

next call: 'system(whoami)'.
    Herbertmayer
```

- #include<bits/stdc++.h>
  using namespace std;
  void system1(string s) {
    const char*command=s.c_str();
    system(command);
  }
  int main() {
    system1("ls");
    system1("pwd");
    system1("whoami");
  }
- ls is not recognized as an internal or external command, operable program or batch file. Pwd is not recognized either as an internal or external command, operable program or batch file. We can see that ls and pwd commands are not recognized in my computer and only whoami is functional, whoami outputs the pc name and its user account that is woking.

6. Write the C or C++ program ***system2*** that reads, when being executed, OS commands via the "command line parameter" C/C++ feature, and then **executes them**. The command line parameters must be legal Unix/Linux commands. Print the number of commands entered. To prepare, read about ***argc, argv***, and ***envp***. Focus is only argc and argv. A sample execution by fictitious user "herb" is:

   herb$ ./a.out pwd ls whoami

   argc = 4
   argv[1] = pwd
   Executing command 'pwd'
   /Users/herbertmayer

   argv[2] = ls
   Executing command 'ls'
   Applications Downloads Music familie test1
   Contacts Favorites Pictures herb test1.c
   Desktop Library Public pix_low_2020 test2
   Documents Movies a.out test.c test2.c

   argv[3] = whoami
   Executing command 'whoami'
   Herbertmayer

   - #include <iostream>
     Int main(int argc, char const *argv[]){
       std::cout << "argc "" << argc << '\n';
       for(int I =1; I <argc; i++){
         std::cout << "argv[" <<I << "] = " << argv[i] << '\n';
         std::cout << "Executing command '" << argv[i] << "'\n";
         system(argv[i]);
         std::cout << '\n';
       }
       return 0;
     }
   - ./a.out pwd ls whoami

```
argc 4
argv[1] = pwd
Executing command ‘pwd’
/home/runner/19255

argv[2] = ls
Executing command ‘ls’
a.out main.cpp

argv[3] = whoami
Executing command ‘whoami’
runner
```

7. What does API stand for? What is an API? What does the API define?
    - API stands for Application Programming Interface, the purpose of API is a set of some protocols and the tools for developing the software apps, and API specifies how these software components should interact with each other. API is usually used when the program the GUI meaning graphical user interface components. The API makes easier to develop a program by providing all the building blocks then the work of programmer to put all the program blocks together to get the desired output.

8. What are 5 general key functions and responsibilities of an OS?
    - User interface, resource management, task management, file management, utilities.
    - Manage the computer’s resources, such as the ventral processing unit, memory disk drives, and printers.
    - Establish a user interface and execute and provide services for applications software.

9. OS commands can be purely textual, or completely graphical. Explain pros and cons. Name a sample OS for both types.
    - CLI commands must be memorized in order to execute various operations.
    - It necessitates more user flexibility.
    - CLI is faster to process since it does not require visuals. As a result, no loading time is required.
    - CLI does not allow for multitasking. Only a few pieces of software have this capability. In contrast, all graphical user interfaces (GUIs) enable multitasking.
    - The user has complete control over the file system when using the CLI. As a result, it is a superior alternative when greater control over the file system and operating system is necessary.
    - Example of CLI - MS DoS, Unix.
    - There is no need to memorize the GUI instructions. The user only needs to recognize the options and work accordingly.
    - The graphical user interface is more user pleasant.
    - CLI takes longer to process than CLI since visuals are employed, which take time to load.
    - GUIs allow multitasking, which means that many tasks may be completed at the same time.

- When compared to CLI, there is less control over the file system.
- The resulting answer is readily apparent, and detecting mistakes is simplified in the GUI.
- Example of GUI - Windows, Linux, Android.

10. What is the meaning of ***Information Hiding***? How is this related to OS?
    - Information hiding is the process of hiding the details of an object or function. The hiding of these details results in an abstraction, which reduces the external complexity and makes the object or function easier to use. In addition, information hiding effectively decouples the calling code from the internal workings of the object or function being called, which makes it possible to change the hidden portions without having to also change the calling code. Encapsulation is a common technique programmers use to implement information hiding. In object-oriented programming, information hiding (by way of nesting of types) reduces software development risk by shifting the code's dependency on an uncertain implementation (design decision) onto a well-defined interface.

11. Briefly contrast ages, origins, uses of: MS-DOS, Unix BSD, Linux
    - MS-DOS (Microsoft-Disk Operating System) is a 16 bit OS that's available freely and requires codes to operate. The first version was in 1981 and latest version is 1995. Its size is very small due to which it is easier to put and its very lightweight.
    - UNIX works as a multitasker and is very much stable. The main use of UNIX is in mainframe computers and in workstations. The first version was released in 1971 and latest was in 1989. It supports multitasking feature and it's a portable OS.
    - LINUX is also available freely and its based on kernel concept, it has similar characteristics like windows and it is one of the safest OS. The latest version was in 2006.

12. Write a concise, ***complete*** English description of the C/C++ function **system().** Be precise and thorough.
    - The C/C++ standard library includes the system() function and its used to pass commands that can be executed in the operating system's command processor or terminal, and it then returns the command once it has been completed. For this method to be called you have to include <stdlib.h> or <cstdlib> and for the syntax int system(char command) if this command is executed without problems then this method returns zero.