Universidad Católica San Pablo **Departamento de Ciencia de la Computación**

Curso: Sistemas Operativos

2022-II

Grupo: CCOMP6-1
Fecha: _____
Profesor: Julio Santisteban
Apellidos y Nombre del alumno_____

1) Familiarizarse con Visual Studio Code y Ubuntu
   En casa usaran 1) ubuntu doble partición o 2) ubuntu WSL
2) Familiarizarse con los mecanismos de compilación en la línea de comando

   FILE: hw.c

```
/* header files go up here */
/* note that C comments are enclosed within a slash and a star, and
may wrap over lines */
// if you use gcc, two slashes will work too (and may be preferred)
#include <stdio.h>
/* main returns an integer */
int main(int argc, char *argv[]) {
/* printf is our output function;
by default, writes to standard out */
/* printf returns an integer, but we ignore that */
printf("hello, world\n");
/* return 0 to indicate all went well */
return(0);
}
```

Compilation and Execution

```
prompt> gcc hw.c
prompt> ./a.out

prompt> gcc -o hw.exe hw.c
prompt> ./hw.exe
```

Useful Flags

```
prompt> gcc -o hw hw.c # -o: to specify the executable name
prompt> gcc -Wall hw.c # -Wall: gives much better warnings
prompt> gcc -g hw.c # -g: to enable debugging with gdb
prompt> gcc -O hw.c # -O: to turn on optimization
```

Linking with Libraries
Note that fork() is a system call, and not just a library routine. However, the C library provides C wrappers for all the system calls, each of which simply trap into the operating system.

```
#include <math.h>
...
tan()
...

// Link with -lm.

prompt> gcc -o hw.exe hw.c -Wall –lm
```

The -lXXX flag tells the linker to look for libXXX.so or libXXX.a,
probably in that order.

if you want the compiler to search for headers in a different path than the usual places, or want it to link with libraries that you

specify, you can use the compiler flag -I/foo/bar to look for headers in the directory /foo/bar, and the -L/foo/bar flag to look for libraries in the /foo/bar directory.

-L/foo/bar;/foo/bar333;/foo/barv55

The -c flag tells the compiler just to produce an object file — in this case, files called hw.o and helper.o. These files are not executables, but just machine-level representations of the code within each source file. To combine the object files into an executable, you have to “link” them together; this is accomplished with the third line gcc -o hw hw.o helper.o).

3) Tarea, Investigar e implementar el uso de Makefile

Revisar:
Laboratory: Tutorial
F.6 Makefiles

Universidad Católica San Pablo **Departamento de Ciencia de la Computación**

Curso: Sistemas Operativos

2022-II

Los trabajos se entregarán usando incluyendo un makefile