Creating and using libraries

Data and File Structures Laboratory

Types of libraries

- Static libraries (`.a` for *archive*): library becomes part of the executable
- Dynamically linked shared object libraries (`.so` for *shared object*): library is not included in the executable, but functions are called as needed at runtime

Creating / using static libraries

- Compiling: `gcc -c abc1.c abc2.c`
- Creating the library: `ar -c libabc.a abc1.o abc2.o`
  - Other options (see man page for more details):
    - `c`: *create*
    - `q`: *quick* (append)
    - `r`: *replace*
    - `u`: *update*
    - `v`: *verbose*
    - `t`: *list*
- Using the library:

```
gcc -o prog prog.c libabc.a
```

or

```
gcc -o prog prog.c -L/path/to/library-directory -labc
```

Creating shared object libraries

position independent code

- Compiling: `gcc -Wall -fPIC -c *.c`

these options are passed to the linker

- Creating the library:
  `gcc -shared -Wl,-soname,libabc.so -o libabc.so *.o`
- Using the library:
  `gcc -o prog prog.c -L/path/to/library-directory -labc`

Using libraries

Compiler flags

- `-I`: specify additional directories to search for header files
- `-L`: specify additional directories to search for library files

Using libraries

Compiler flags

- `-I`: specify additional directories to search for header files
- `-L`: specify additional directories to search for library files

Environment variables

- `export C_INCLUDE_PATH=/path/to/header-files`
- `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/path/to/lib`

Source

- http://www.yolinux.com/TUTORIALS/LibraryArchives-StaticAndDynamic.html