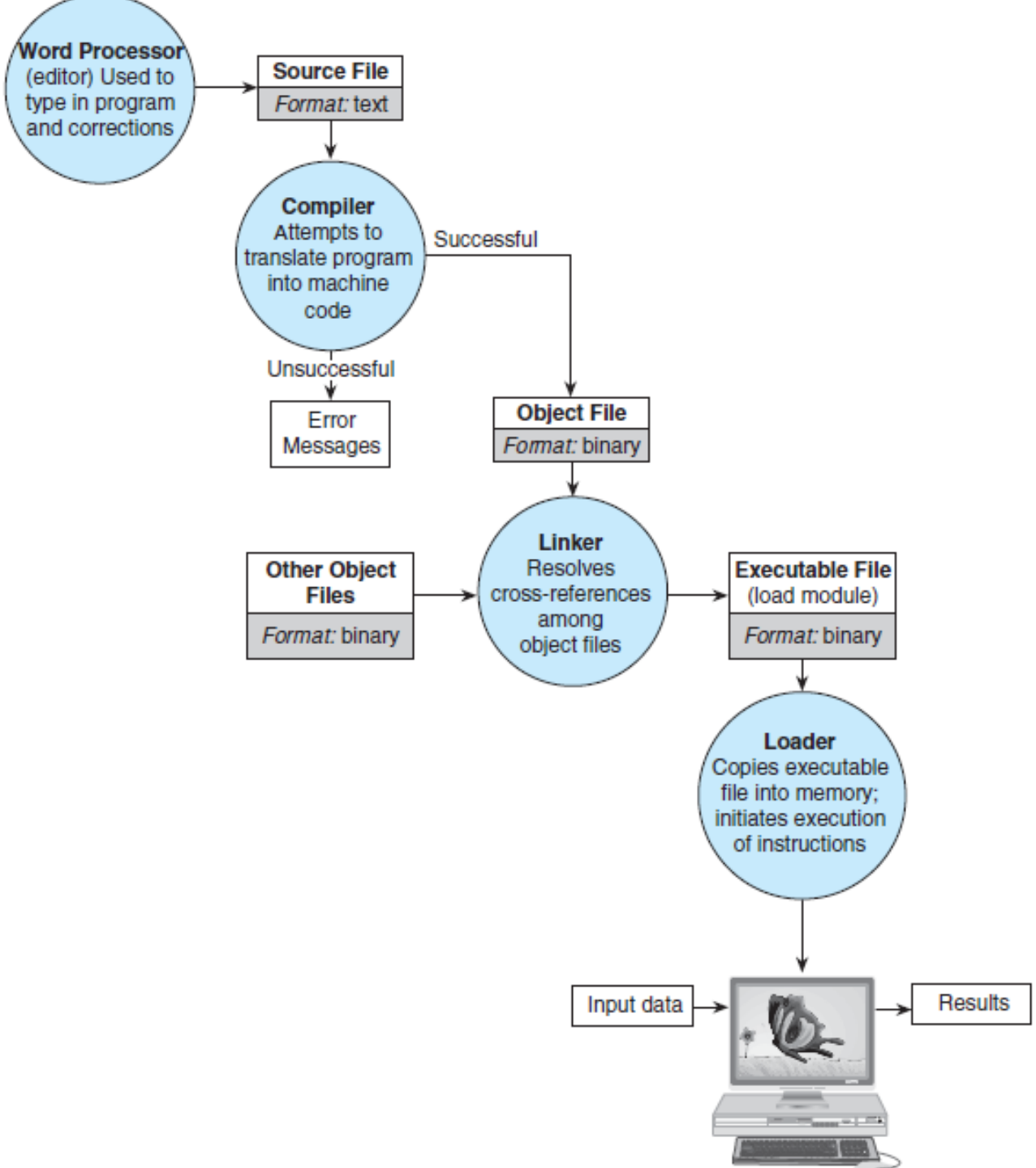
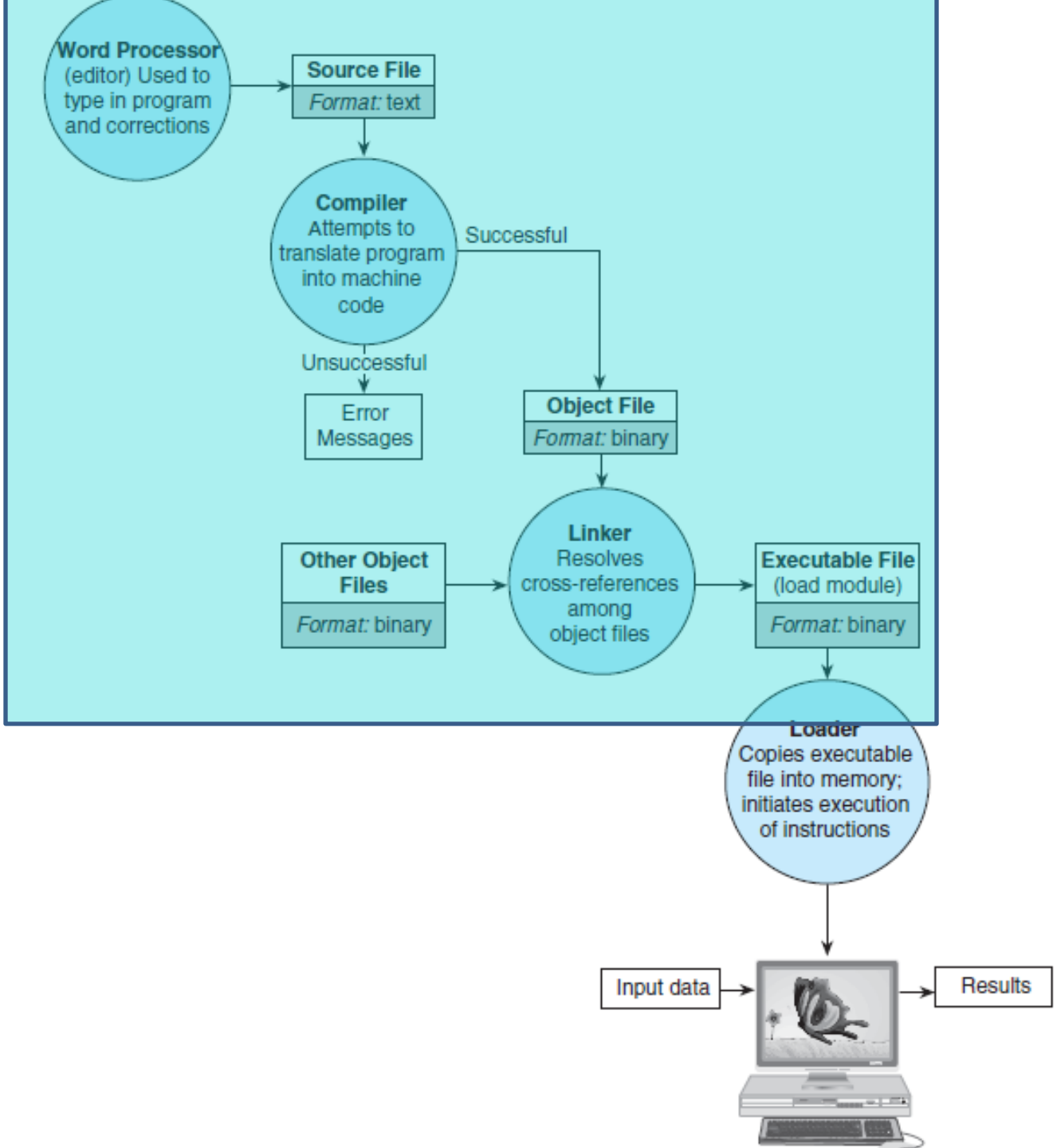
CSCI 112
Final review


Word Processor
(editor) Used to type in program and corrections
Source File
Format: text
Compiler
Attempts to translate program into machine code
Successful
Unsuccessful
Error Messages
Object File
Format: binary
Other Object Files
Format: binary
Linker
Resolves cross-references among object files
Executable File
(load module)
Format: binary
Loader
Copies executable file into memory; initiates execution of instructions
Input data
Results

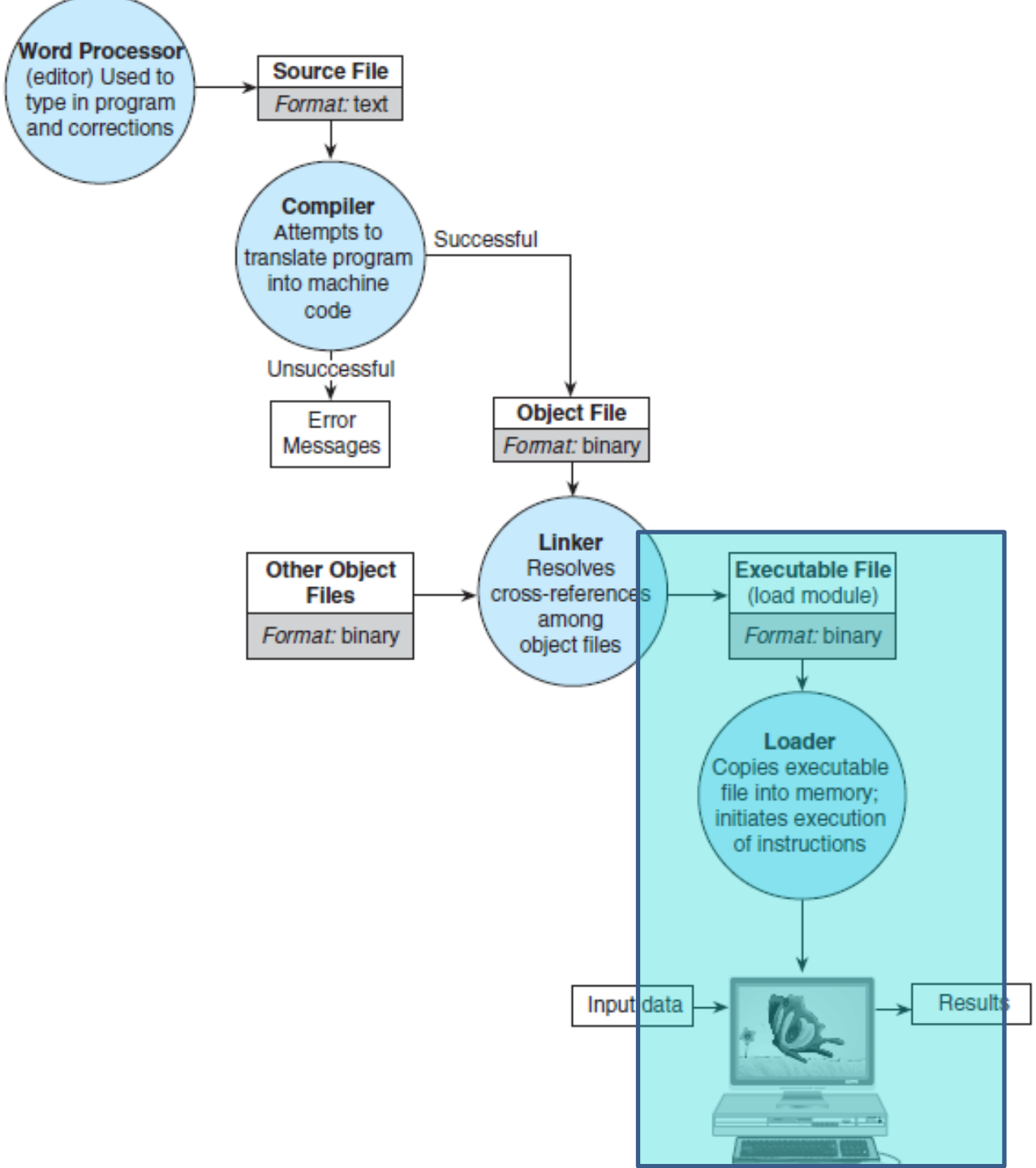

Word Processor
(editor) Used to type in program and corrections
Source File
Format: text
Compiler
Attempts to translate program into machine code
Successful
Unsuccessful
Error Messages
Object File
Format: binary
Other Object Files
Format: binary
Linker
Resolves cross-references among object files
Executable File
(load module)
Format: binary
Loader
Copies executable file into memory; initiates execution of instructions
Input data
Results

Creating an executable

gcc –o exe myprogram.c –Wall

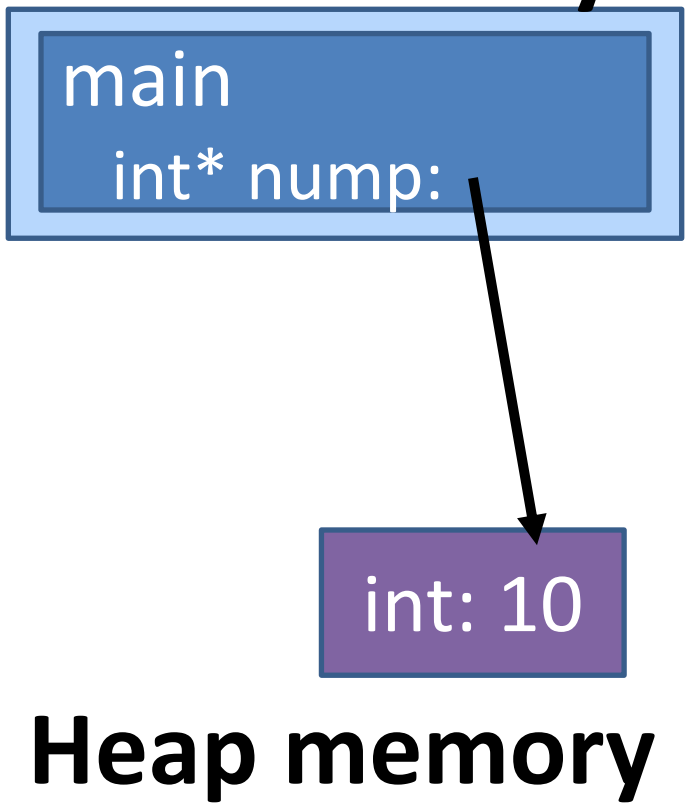
Makefile

```
lab7: lab7.o funcs.o
        gcc lab7.o funcs.o -o lab7

lab7.o: lab7.c lab7.h
        gcc -c lab7.c -Wall

funcs.o: funcs.c lab7.h
        gcc -c funcs.c -Wall
```


Word Processor
(editor) Used to type in program and corrections
Source File
Format: text
Compiler
Attempts to translate program into machine code
Successful
Unsuccessful
Error Messages
Object File
Format: binary
Other Object Files
Format: binary
Linker
Resolves cross-references among object files
Executable File
(load module)
Format: binary
Loader
Copies executable file into memory; initiates execution of instructions
Input data
Results

What happens when we run our executable file?


```
int main(void) {
    int* nump;
    nump = malloc(sizeof(int));
    *nump = 10;
    free(nump);
}
```



Reading in data

Method of input	What C program looks like	How to run executable
Redirection	scanf("%d", &myint); scanf("%s", myword);	./exe < input.txt

Open file in C	FILE* myfile;
	myfile = fopen("input.txt", "w")
	fscanf(myfile, "%d",&myint);
	fgets(line, 100 myfile);
		./exe

Command line argument	int main(int argc, char* argv[]) {
	int num_args = argc – 1;
	char* first_arg = argv[1];
		./exe FirstArg SecondArg

Passing by value vs. passing by reference

- Arrays and strings are always passed by reference
- Ints, chars, doubles, pointers are passed by value, but you can pass in pointers to them to pass by value

Let’s look at values_vs_reference.c