Chapter 1

Hello World

Write a program that uses write() to print out “Hi my name is “.

```
#include <unistd.h>

int main() {
   write(1, "Hi my name is Abhishek", 22);
   return 0;

}
```

Hello Standard Error Stream

Write a program that uses write to print out a triangle of height n

```
#include <unistd.h>

#define STDOUT_FNO 1
#define STDOUT_FILENUM 2
int main() {
   int len;
   for(len = 3; len; len--){
      write(STDOUT_FNO, "***", 3 - len + 1);
      write(STDOUT_FILENUM, "\n", 1);
   }
   return 508;

}
```

Writing to Files

Write “Hello World” to a file

```
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main() {
   //int count;
   mode_t mode = S_IRUSR | S_IWUSR;
   int files = open("output.txt", O_CREAT | O_TRUNC | O_RDWR, mode);
   write(files, "Hello World\n", 12);
   close(files);
   return 42;
}
```

Not everything is a system call

Use printf to to write to a file

```
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>

int main() {
  //int count;
  mode_t mode = S_IRUSR | S_IWUSR;
  close(1);                                    // this closes the standard out path causing
everything after to print to the next outputline.
  int files = open("output.txt", O_CREAT | O_TRUNC | O_RDWR, mode);
  printf("Hello World ");
  write(files, "Hello World\n", 12);
  close(files);

  return 42;

}
```

Name some difference between write() and printf()

write() requires you to know how many bit are going to be used and will print only that much out where printf() does not need that. Also printf() is a buffer so unless you have a special character such as \n in the contents printf() will always output after anything in the write() regardless of where in the code it is.

Chapter 2

Not all bytes are 8 bits

How many bits are there in a byte?

At least 8 bits

How many bytes are in char?

1 byte a character has 8 bits in this architecture.

Tell me how many bytes the following are on your machine:

int – 4 bytes
double – 8 bytes
float – 4 bytes
long – 4 bytes
long long – 8 bytes

If the address of data is at 0x7fbd9d40, then what is the address of data + 2

Data + 2 is at 0x7fbd9d48 this is the same as data[2] so data[3] would be at 0x7fbd94c

Remember the type of string constant “abc” is an array

Why does the following code have a segmentation fault?

Char* ptr = “hello”;
*ptr = ‘J’;

This is because there are parts in memory that you can and can not change and hello in this case is constant so therefore you can’t change ptr and thus you get a segmentation fault for doing so.

What does sizeof(“Hello\0World”) return?

It returns 12

What does strlen(“Hello\0World”) return?

It returns 5

Give an example of X such that sizeof(X) is 3

X = “He”

Give an example of Y such that at sizeof(Y) might be 4 or 8 depending on the machine

Y =

Chapter 3

Name me two ways to find the length of argv

One way is to get have a while loop until you reach the 0 bit of the argv value and have a counter that increments through each iteration.

What is argv[0]

Argv[0] is the first input that you can give into the program.

Where are the pointers to the environmental variables stored

They are stored in the root directory of the machine.

String searching

What is the results of sizeof(ptr) and sizeof(array)

Char* ptr = “Hello World”; = 4 – due to the pointer not being dereferenced and that is the size that it takes up in the machine

Char array[] = “Hello World”; = 12 – due to everything being dereferenced so it will count for everything.

What data structure is managing the lifetime of automatic variables?

They are managed in a heap

Chapter 4

Memory allocation using malloc, heap and time

If I want to use data after the life time of the function it was created in, then where should I put it and how do I put it there?

You should put the variable outside all of the functions and use the keyword static upon declaration. You can also use malloc to reserved a certain amount of memory ahead of time in case you run out.

Fill in the blank

For every malloc there is a **free**

Heap allocations Gotchas

Name one reason malloc can fail

If you program uses up all of the heap memory

Name some difference between time() and ctime()

Ctime uses static storage. Also what is being returned by the two methods are different. One of them returns a variable of time_t while the other returns a character pointer for time in ASCII values

What is wrong with this code snippet?

```
Free(ptr);
Free(ptr);
```

The pointer has been freed up twice one after the other and will cause a segmentation fault because the pointer has been freed up.

What is wrong with this code snippet?

```
Free(ptr);
Printf("%s\n", ptr)
```

The pointer has been freed up and is now pointing to null so it will print out null since you are asking for the string value

How can one avoid the previous 2 mistakes?

After you free the pointer set that pointer to null. This solves the dangling pointer issue.

Struct, typedefs, and a linked list

Create a struct that represents a Person and typedef, so that "struct person" can be replaced with a single world

```
Struct Person {
        Char* name;
        Int* age;
        Struct person*[] friends;
}

typedef struct Person person_t;
```

Now make two persons "Agent Smith" and "Sonny Moore" on the heap who are 128 and 256 years old respectively and are friends with each other

```
Person_t* As = (person_t*) malloc(sizeof(person_t));
Person_t Sm = (person_t*) malloc(sizeof(person_t));
As->name = "Agent Smith";
Sm->name = "Sonny Moore";
As->age = 128;
Sm->age = 256;
As->friends[0] = Sm;
Sm->friends[0] = As;
Free(As);
Free(Sm);
```

Duplicating strings, memory allocation and deallocation of structures

```
Person_t* person_create(char* aname, int* aage) {
        Person_t* result = (person_t*)Malloc(sizeof(person_t));
        Result -> name = aname;
        Result -> age = aage;
        Return result;
}
void link_destroy(person_t *p) {
        free(p->name);
        free(p->age);
        memset(p, 0, sizeof(person_t));
        free(p);
}
```

Chapter 5

Reading characters, Trouble with gets

What functions can be used for getting characters for stdin and writing them to stdout?

You can use getchar() and putchar() in a loop if given the argument.

Name one issue with gets()

You can run into a bufferoverflow issue and that can change the value of whatever you are manipulating.

Introducing sscanf and friends

C Development

What compiler flag is used to generate a debug build?

Gdb is the best flag as it gives stack based error message.

You modify the makefile to generate debug builds and type make again. Explain why this is insufficient to generate a new build

This is because when you build a new part after modifying code if you do not build everything again there will be a version issue and the program will most likely not perform the way you want it to.

Are tabs or spaces used in Makefiles?

Spaces are used but not tables to separate the various flags.

What are the differences between heap and stack memory?

Heap is more permanent memory where as the stack increases as the program needs it and mainly contains locally declared variables

Are there other kinds of memory in a process?

There is physical memory and virtual memory