Conceptual Questions

1. What are benefits of C, where is it used

C language is a middle level language, meaning that it can be used for hardware level applications and software applications. C is also a very readable language considering how low level it actually is. C was used to make the Linux operating system as well as most driver software for modern computers. C is also used for most embedded software such as micro controller firmware.

2. What is a compiler, what does it do?

The compiler is program that takes code written in a spcific syntax and converts it to machcine code which can be executed by a specific processor. Compilers are language and architecture specific. This means that there will be a different C compiler for each type of architecture. The compiler will parse through the code checking for syntax errors. It will then tokenize the code into individual parts. Then, the compiler will attempt to derive the symmantics of these parts and it will attempt to optimize the code for the specific architecture. Then, the required libraries as well as other required files will be bundled and converted to machine code as an executable.

3. What is a Makefile, what does it do?

A make file is a simple way to associate short names called targets to a set of scripts, usually used to call compilation commands when building software. In order to use makefiles, the Makefile program must be installed. While in the containing directory, the user simply types 'make' and the makefile will go through its commands and compile the required files. This is a fast way to build software so that other users can quickly build and test it without compiling each individual file.

4. Name 5 header files from C library, explain their purpose

stdlib.h

General utilities for writinng c programs including memory management, program utilities, string conversions and random values.

stdio.h

Input and output utilities for reading and writing to files and command line.

string.h

String utilities including case changing, comparing and copying of strings.

time.h

Clock utilities for timing events and processes, calculating time and formatting based on common

time formats.

signal.h

Provides utilities to handle signals.

5. Look up one of the functions from each header and describe what it does

stdlib.h - void* malloc(size_t size)

Allocates memory based on amount specified by 'size'.

stdio.h - int printf(const char *format, ...)

Writes string to stdout based on string template by replacing arguments.

string.h - char *strcpy(char *dest, const char *src)

Copies a null terminated string pointed to by src to the destination pointed to by dest.

time.h - double difftime(time_t time_end, time_t time_beg)

Computes the difference between two calender times as time_t objects in seconds.

signal.h - void abort(void)

Causes abnormal program termination unless SIGABRT is caught and handeled by a different signal handeler.

Application Questions

1.

```
int main(void) {
    int ar[10]; // declare array
    srand(time(NULL)); // seed random number generator
    int i;
    for (i=0; i<10; i++) {
        ar[i] = rand()%10; // generate random number for each array element
        printf("%d ", ar[i]); // print the current array element
    }
    printf("\n"); // print line break
}
```

Output

```
1 9 2 0 0 8 3 6 4 3
```

2.

```
int main() {
    float ar[] = {1.2, 5.5, 2.1, 3.3, 3.3};
    int i;
    // start at 1 becase no elements before index 0
    for (i=1; i<5; i++) {
        if (ar[i] > ar[i-1])
            printf("greater than\n");
        else if (ar[i] < ar[i-1])
            printf("less than\n");
        else
            printf("the same\n");
    }
}
```

Output

```
greater than
less than
greater than
the same
```

3.

```
int main() {
    char str[] = "hello world";
    int i = 0;
    while (1) {
        if (str[i] == '\0') // check for null terminator
            break;
        printf("%c", str[i]);
        i++;
    }
    printf("\n");
}
```

Output

```
hello world
```

4.

```
int main() {
```

```
    int i;
    for (i=1; i<=10; i++) {
        printf("%s ", (i%2==0) ? "even" : "odd");
    }
    printf("\n");
}
```

Output

```
odd even odd even odd even odd even odd even
```

5.

```
// calculate the distance between 2 points
double euclid_dist(int x1, int y1, int x2, int y2) {
    return sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2));
}

int main() {
    int i;
    for (i=0; i<10; i++) {
        srand(time(NULL)+i);
        // random values (0, 100) for points
        int x1=rand()%100, y1=rand()%100, x2=rand()%100, y2=rand()%100;
        // print distance between 2 points
        printf("V1=(%d, %d) V2=(%d, %d) D=%f\n", x1, y1, x2, y2, euclid_dist(x1, y1,
x2, y2));
    }
}
```

Output

```
V1=(45, 8) V2=(98, 7) D=53.009433
V1=(26, 16) V2=(88, 65) D=79.025312
V1=(61, 82) V2=(4, 29) D=77.833155
V1=(60, 67) V2=(8, 44) D=56.859476
V1=(46, 24) V2=(20, 39) D=30.016662
V1=(37, 32) V2=(18, 69) D=41.593269
V1=(44, 43) V2=(61, 44) D=17.029386
V1=(90, 3) V2=(56, 80) D=84.172442
V1=(14, 87) V2=(58, 19) D=80.993827
V1=(56, 89) V2=(61, 33) D=56.222771
```