CIS 2107
Assignment 1
Written Part

1 Tools

1. We've said that when we run gcc on a .c file, there are several stages: preprocessing, compiling, assembling, and linking.

   (a) What happens during the preprocessor stage? What is input? What is output?

   input: c file. Processes preprocessor commands such as the ones that start with #. Copies included files' methods and references into output as well. Does various other preprocessing and outputs a c file

   (b) What happens during the compilation stage? What is input? What is output?

   input: preprocessed c file. Code is converted to assembly language. This step may as well be skipped for faster compiling time

   (c) What happens during the assembly stage? What is input? What is output?

   input: assembly code. Assembly code is converted into machine code, which is in binary format.

   (d) What happens during the linking stage? What is input? What is output?

   input: machine code. Machine code has instructions that the computer can interpret and run, but it has missing instructions, that are those of included libraries'. After the linking stage an executable file is generated, which can be natively run by computer

2. What's the command to "go up" a level in the directory tree?

   2. cd ..

3. What's the command to "go to" your home directory?

   3. cd ~

4. What's the command to "go to" the root of the directory tree?

4. cd /

5. What's the command to delete a file?

5. rm filename

6. What's the command to delete a directory?

rmdir dirname

if the directory is not empty: 6. rm -rf dirname

7. What's the command to create a directory?

7. mkdir dirname

8. What's the command to make a copy of a file?

cp source destination

9. What's the command to rename a file?

mv filename newname

10. Suppose that there's a unix shell command called `petronam`. What's the command that I'd type in the shell in order to see the help page for the proper usage of `petronam`?

for manual page: man petronam if the program supports, for help: petronam --help

11. What's the unix shell command to view the contents of a file in hexadecimal?

hexdump filename

12. What's the unix shell command to display a file one screen at a time?

cat filename actually there are many ways. Like less filename, and nano filename, or vi filename etc

13. How would I run the command `keurig` taking its input from a file called `beans`?

depending on your use case: cat beans | keurig or keurig $(cat beans)

14. How can I take the output of a `bake` command and use it as input for a `slice` command?

depending on your use case: bake | slice or slice $(bake)

15. What's the command to print a long directory listing (names, permissions, owner, length, *etc*)?

ls -l

16. What's the command to alter the permissions of a file to allow the owner to read, write, or execute the file?

chmod 700 filename

2 Arguments to Functions

17. What's printed by each of the following?

(a)
```
public class WhatsPrinted01 {
    public static void func(int x) {
        x++;
    }

    public static void main(String args[]) {
        int x=5;
        func(x);
        System.out.println(x);
    }
}

```

(a) 5

(b)
```
public class WhatsPrinted02 {
    public static void func(int x, int y, int z) {
        System.out.println("x=" + x +
                           ", y=" + y +
                           ", z=" + z);
    }

    public static void main(String args[]) {
        int x=5, y=10, z=15;
        func(y, z, x);
    }
}

```

(b) x=10, y=15, z=5

(c)
```
public class WhatsPrinted03 {
    public static void func(int x, int y, int z) {
        x++;
        y+=x;
        z+=y;
    }

    public static void main(String args[]) {
        int x=5, y=10, z=15;
        func(y, z, x);
        System.out.println("x=" + x +
                           ", y=" + y +
```

```
                                ", z=" + z);
    }
}
```

(c) x=5, y=10, z=15

(d)
```
public class WhatsPrinted04 {
    public static void func(int A[]) {
        A[0]++;
    }

    public static void main(String args[]) {
        int A[] = {10,20,30};
        func(A);
        System.out.println(A[0]);
    }
}
```

(d) 11

(e)
```
public class WhatsPrinted05 {
    public static void func(int A[]) {
        int B[] = new int[A.length];
        for (int i=0; i<A.length; i++)
            B[i]=A[i]+1;
        A=B;
    }

    public static void main(String args[]) {
        int A[] = {10,20,30};
        func(A);
        System.out.println(A[0]);
    }
}
```

(e) 10

(f)
```
class Derp {
    public int x;

    public Derp(int x) {
        this.x=x;
    }

    public String toString() {
        return new String("x=" + x);
    }
}

public class WhatsPrinted06 {
    public static void func(Derp d) {
        d.x++;
        System.out.println(d);
    }

    public static void main(String args[]) {
        Derp d1 = new Derp(10);
        func(d1);
    }
}
```

(f) x=11

(g)
```
class Derp {
    public int x;

    public Derp(int x) {
        this.x=x;
    }

    public String toString() {
        return new String("x=" + x);
    }
}

public class WhatsPrinted07 {
    public static void func(Derp d) {
        d.x++;
    }

    public static void main(String args[]) {
        Derp d1 = new Derp(10);
        func(d1);
```

```
21          System.out.println(d1);
22      }
23  }
```

(g) x=11

(h)
```
class Derp {
    public int x;

    public Derp(int x) {
        this.x=x;
    }

    public String toString() {
        return new String("x=" + x);
    }
}

public class WhatsPrinted08 {
    public static void func(Derp d) {
        d = new Derp(222);
    }

    public static void main(String args[]) {
        Derp d1 = new Derp(10);
        func(d1);
        System.out.println(d1);
    }
}
```

(h) x=10

3 Thinking about the memory hierarchy

In class, we discussed the computer storage hierarchy, and the fact the types of storage high in the "pyramid" given in Chapter 1 (*e.g.* registers, L1 cache, ...) are orders of magnitude faster than the types at its base (*e.g.* hard drives, optical storage, ...).

Some access times are given for current hardware[1] . (Recall that the abbreviation ns is for nanoseconds and ms is for milliseconds.)

So, looking at the table, we see for example if we'd like to read something stored in a L1 cache, it should take about a half of a nanosecond to fetch it, *etc.*

location	**access time**
L1 cache	0.5 ns
L2 cache	7 ns
RAM	100 ns
hard drive	10 ms
DVD	140 ms

In our every day lives, we're not used to dealing with units of time this small. To give us a sense of how much slower devices at the bottom of the pyramid are from the top, let's pretend instead that access times are in units to which we're more accustomed.

Using the same ratios as we had in the table above, fill in the following *fake* access times. For example, in the table of real access times, we see that it takes 0.5 ns on average to read from a L1 cache, and 7 ns to read from a L2 cache. L2 caches are on average 14 times slower. If we pretend that it takes 1 second to read from an L1 cache, it would take 14 seconds to read from a L2 cache. Using this same procedure, fill in the rest of the table.

location	***fake access time***
L1 cache	1 sec
L2 cache	14 sec
RAM	200 sec
hard drive	20 000 000 sec
DVD	280 000 000 sec

[1] Most of these come from a talk given at a recent conference on systems http://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf.

Now, let's do the same thing, but instead of time, let's consider distance. Pretend that we're baking a cake. We're doing all of our work on the kitchen counter, and different ingredients that we need for the cake are found in different storage places. If the L1 cache is the cabinet two feet above the counter, the L2 cache would be 14 times farther away, so it's 28 feet away (maybe in our basement?). Using the same ratios as before, fill in the rest of the chart:

location	***fake distance***
L1 cache	2 feet
L2 cache	28 feet
RAM	133 yard
hard drive	3787 miles
DVD	53030 miles