Computer Science 1MC3

Lab 9 – Coding Practices

When coding a program there are a few conventions that help make code more readable by other people. After all, in industry, and at school, it may be necessary for other people to understand your code.

Tabing

Tabing is the process of aligning the code of a control structure a tab space in. Like:

```
int main {

        int x, y, z;

        while(x>y) {
                for(z=1;z<x;z++) {
                        x:=y/x;
                }
        }

}
```

You should always be tabing your code. Not only does it make it more readable it is pretty much common practice among computer scientists.

Comments

Eventually you will be programming code that will be several thousands of lines long. Something that you will learn very quickly is that code which made complete sense at the time of coding will be incomprehensible a year later. What you should do to prevent this is to put comments in your code. Comments are basically little reminders put in your code to clarify or remind you about something. Comments will be skipped over by the complier.

To do a single line comment you do `//your comment` or to do a multiple line comment you do `/** your comments **/`. In most compilers whatever you put in your code will change colour.

Homework

1. Create a function to take a string and a number, x, which returns an array of the string repeated x times:

   ```
   char* wordDuplicate (char *word, int x);
   ```

2. Using malloc declare two character arrays on the stack. Create a program to go out of bounds on your first array until it hits the second array. Use the string `"chk"`, you may assume that if you find this string on the heap that it isn't a coincidence

3. Create a program to take a sentence and output the following:

   a. a chart of the word frequency
   b. the first and the last word

c. the longest word
d. how many characters are in the string

4. *Difficult*

Create a program to take an array of consecutive integers of any length [1,2,3. . n] and produce a list of all subsets of length 3 of this list in lexica graphica ordering.

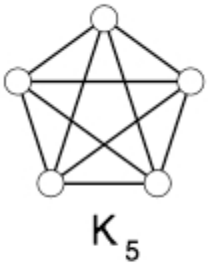
For example

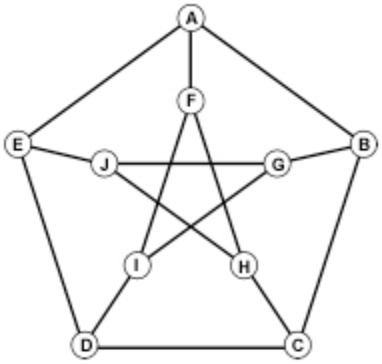
A=[1,2,3,4,5,6]

would produce

123
124
125
126
134
135
136
145
146
156
234
235
236
245
246
256
345
346
356
456

4. *very difficult* A Hamilton graph can looks like this:





The circles are **nodes** and the lines between them are called **edges**. A legal “move” in a graph is to travel from one node to another along an edge (a can go to b, f or e). A **Hamilton path** is a group of movements in which every node is visited exactly once. If you were to label the nodes with the letters a-b-c-d-e-f-g-h-i-j, you could describe a path like: a-b-c-d-e-j-h-f-i-g.

Furthermore, a Hamilton path that ends with a node that is a neighbor of the node you started with is considered a **Hamilton circuit**.

Create a program to find ALL Hamilton circuits.