Converting to “Goto C”

Pseudocode	Goto C

`x = 0`
`while (x < 6)`
`    print(x)`
`    x = x + 1`	

`if (i==j||i==k)`
`    i=i+1`
`else`
`    j=j-1`
`j=i+k`	

`n = 1`
`fact = 6`
`for(i = 1; i <= fact; i++)`
`    n = n * i`
`return n`	

Bonus!

```
int binary_search(int N)
{
      int found = 0; int min=0; int max=N-1; int index=-1;
      while (!found){
            if (min > max)
                  break;
            else {
                  index = floor((min+max)/2);
                  if (key == A[index])
                        found = 1;
                  else if (key < A[index])
                        max = index - 1;
                  else
                        min = index + 1;
            }
      }
      if(found == 1)
            return index;
      else
            return -1;
}
```

Linux Cheat Sheet

`ls`: list everything in your current directory
`cd`: change your directory (default to ~, your home)
- **`cd` *`path`***: change to a path (ex. `cd Documents/myStuff`)
- **`cd ..`** : change to directory before your current directory
- **`cd -`** : undo last cd
- (You can combine: `cd ../../Documents/myStuff`, if you were in `Pictures/myStuff`)

`pwd`: print working directory (where am I?)
`mkdir` *`dir`*: create a directory/folder
`rm`: remove
- **`rm` *`file`***: remove file *file*
- **`rm -r` *`dir`***: remove directory *dir*

`more` *`file`*: output *file* to the screen
`vi` *`file`*: open *file* in the text editor vi (vi commands below)
- **`:I, :i`**: enter insert mode
- **`esc`**: exit a mode
- ***`line #`*** **`G`**: jump to a line number
- **`gg`**: jump to the beginning of the file
- ***`/thing`***: search for phrase *thing* in file, use **`n`** to jump to the next occurance
- **`:wq`**: save and quit
- **`:q`**: quit
- **`:q!`**: force quit

`./ex`: run executable *ex*
- ***`./ex < stuff.txt`***: run executable *ex* and take input from file *stuff.txt*
- ***`./ex > stuff.txt`***: run executable *ex* and output to file *stuff.txt*

** What's hex? I don't remember!*

- ***Hex**, or **hexadecimal**, is the base-16 number system. There are 6 numbers after 9 represented by A, B, C, D, E and F before 10. To indicate that a number is in hex, we preface the number with **0x**. Some examples:*
  - *10 = 0xA*
  - *16 = 0x10*
  - *24 = 0x18*
  - *200 = 0xC8*
- *If you need to convert back and forth, you can use http://www.calculator.net/hex-calculator.html*

*** Dereference? What's that mean?*

- ***Dereferencing** is the act of looking at what is at an address. Say you have register **$eax**, but when you print it, you get a weird number like 4198116, even though you know that something important is supposed to be stored there. It's possible that eax is storing an **address**, meaning that its value is actually just telling you (and the computer) **where** to look for what you actually want. There are two ways to get at the number you want:*
  - ***`x/s $eax`**: Will print whatever is at the address held by eax. **This is the easiest way!!***
  - ***`print *0x400ee4`**: 0x400ee4 is the number 4198116 in **hex**, and is how your computer stores addresses. The asterisk tells print that you want to look at this address for your value.*

Goto C Cheat Sheet

Regular C	**Goto C**

`if (Test)`
`  then-statement;`
`else`
`  else-statement;`
	`if (!Test) goto false;`
	`  then-statement;`
	`  goto done;`
	`false:`
	`  else-statement;`
	`done:`

`while (Test)`
`  Body`
	`loop:`
	`  if (!Test) goto done;`
	`  Body`
	`  goto loop`
	`done:`

`for (Init; Test; Update )`
`  Body`
	`Init;`
	`loop: if (!Test) goto done;`
	`  Body`
	`  Update;`
	`  goto loop;`
	`done:`

`do`
`  Body`
`while (Test);`
	`loop:`
	`  Body`
	`  if (Test)`
	`  goto loop`

Negating an expression: (test ⇔ !test)

- == ⇔ != (equals ⇔ not equal to)
- > ⇔ <= (greater than ⇔ less than or equal to)
- < ⇔ >= (less than ⇔ greater than or equal to)
- && ⇔ || (and ⇔ or)
- ! ⇔ [blank] (get rid of “not”)

Examples:

- !(x < 10) → (x >= 10)
- !(x != 10) → (x == 10)
- !((x > 2) && (x < 20)) → ((x <= 2) || (x >= 20))