

```

PS C:\Users\ludde> I think I'm feeling something.
>>
>>
>> - What?
>> - I don't know. It's strong, pulling me.
>>
>>
>> Bring the nose down.
>>
>>
>> Thinking bee!
>> Thinking bee! Thinking bee!
>>
>>
>> - What in the world is on the tarmac?
>> - Get some lights on that!
>>
>>
>> Thinking bee!
>> Thinking bee! Thinking bee!
>>
>>
>> - Vanessa, aim for the flower.
>>
>>
>> Out the engines. We're going in
I : The term 'I' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify
that the path is correct and try again.
At line:1 char:1
+ I think I'm feeling something.
+ ~

PS C:\Users\ludde> on bee power. Ready, boys?
on : The term 'on' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify
that the path is correct and try again.
At line:1 char:1
+ on bee power. Ready, boys?
+ ~
+ CategoryInfo          : ObjectNotFound: (on:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\ludde>
PS C:\Users\ludde>
PS C:\Users\ludde> Affirmative!
Affirmative! : The term 'Affirmative!' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path
At line:1 char:1
+ Affirmative!
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (Affirmative!:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

```

With a long input the characters will overflow the program and be read through the command line

13/10/20

Functions and Control Flow

<https://secure.ecs.soton.ac.uk/notes/ellabs/1/c2/c2.pdf>

<https://www.arduino.cc/reference/en/language/variables/data-types/unsignedlong/>

https://en.wikipedia.org/wiki/C_data_types

https://en.wikipedia.org/wiki/Printf_format_string

<https://stackoverflow.com/questions/20186809/endless-loop-in-c-c>

An unsigned long variable can be declared like so:

Syntax

```
unsigned long var = val;
```

Parameters

var: variable name.

val: the value you assign to that variable.

To initialize an unsigned long variable I have to make sure the compiler knows what data type I am using. When I declare my variable I must put the type at the end of the initialized value.

```
Unsigned long var = 10UL;
```

The format specifier for an unsigned long variable is %lu.

Fixed width data types are data types that have a specified length, because every system has its own default length. This helps the portability and reliability of the code. To use a fixed width data type I must include the <stdint.h> library, and if using macros the <inttypes.h> library.

When declaring the variable and wanting a fixed character width value I must use the library's data types:

Type category	Signed types			Unsigned types		
	Type	Minimum value	Maximum value	Type	Minimum value	Maximum value
Exact width	int _n _t	INT _n _MIN	INT _n _MAX	uint _n _t	0	UINT _n _MAX
Least width	int_least _n _t	INT_LEAST _n _MIN	INT_LEAST _n _MAX	uint_least _n _t	0	UINT_LEAST _n _MAX
Fastest	int_fast _n _t	INT_FAST _n _MIN	INT_FAST _n _MAX	uint_fast _n _t	0	UINT_FAST _n _MAX
Pointer	intptr_t	INTPTR_MIN	INTPTR_MAX	uintptr_t	0	UINTPTR_MAX
Maximum width	intmax_t	INTMAX_MIN	INTMAX_MAX	uintmax_t	0	UINTMAX_MAX

n is the fixed width of the data type, for example

int_fast8_t

Is the fastest data type of length 8.

Within the printf() and scanf() format strings the fixed width format specifier is %d.

To create an endless loop I can use the for or while commands. I can use the for command to create an endless loop like so:

```
for (;;) {  
    ...  
}
```

is an "infinite" loop, presumably to be broken by other means, such as a break or return. Whether to use while or for is largely a matter of personal preference.

Using the while command to create the endless loop can be done like so:

```
while(1){}
```

However this is likely to be misidentified as an error by the compiler because the statement is always true.

To create a sine function I can use the `sin()` function in the `<math.h>` library. This gives the radian sine value of an input. To increase the frequency I can give the input a coefficient.

The sine value can be scaled to the range of 0 to 1 using the formula $y = (\sin(x) + 1) / 2$

My version of the `plotval` function:

```
1  /*plotval*/  
2  
3  /*libraries*/  
4  #include <stdio.h>  
5  #include <stdint.h>  
6  #include <math.h>  
7  
8  /*definitions*/  
9  #define MAX_TERMINAL 120  
10  
11  /*plotval function*/  
12  int plotval(float ploti) /*Declare plotval function and plot input*/  
13  {  
14      for (int i = 0; i <= roundf(ploti * MAX_TERMINAL); i++) /*Loop till at proportion of the screen to sin value*/  
15      {  
16          printf(" ");  
17      }  
18      printf("\n"); /*Mark with an asterix*/  
19  }
```