What are the following hexadecimal numbers, in **binary** and in **decimal**?

1. 0x64
   1. 100, 1100100
2. 0x1000
   1. 4096, 1000000000000
3. 0x100
   1. 256, 100000000
4. 0x400
   1. 1024, 10000000000
5. 0x1049
   1. 4169, 1000001001001

Each of the numbers in the previous problem can be expressed as a sixteen-bit number. Now that you have written them in binary divide them into bitfields of 6, 5, and 5 bits (with the 6-bit field being the high-order bits).

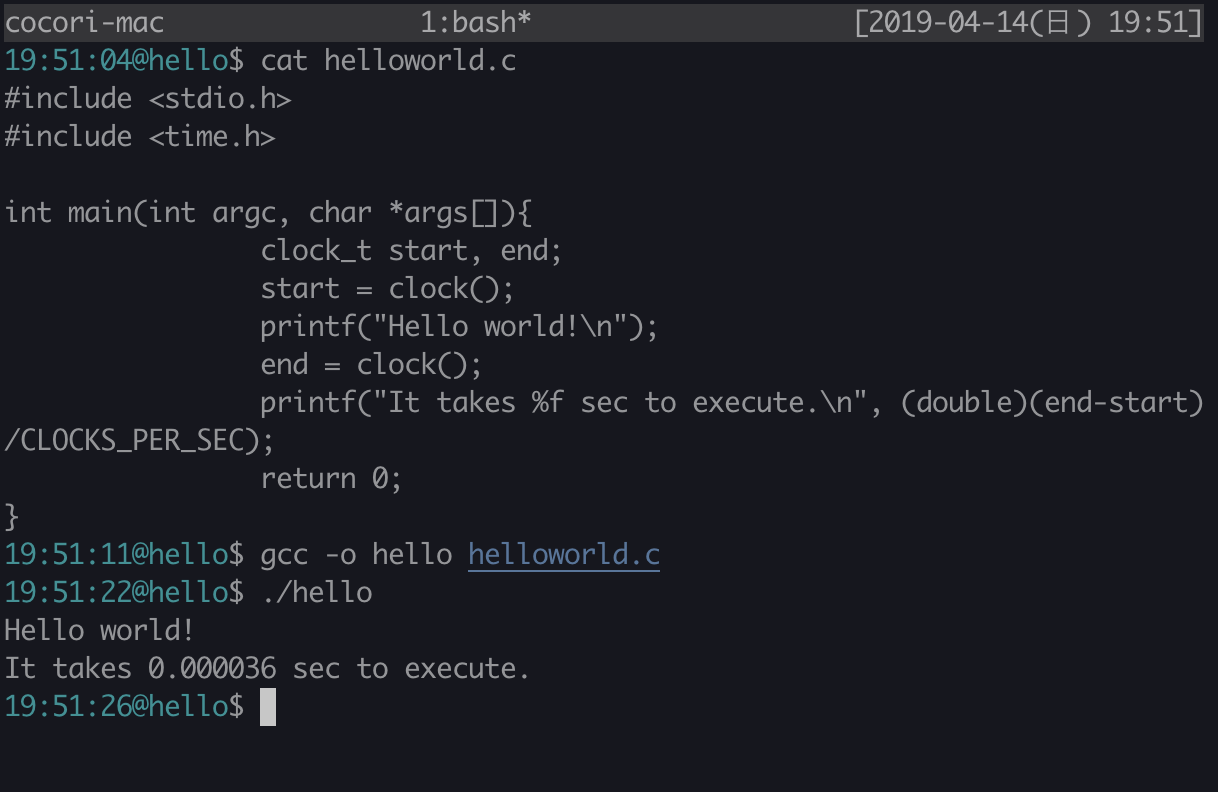
1. 0x64
   1. 100, 1100100
   2. 000000 00011 00100
2. 0x1000
   1. 4096, 1000000000000
   2. 000100 00000 00000
3. 0x100
   1. 256, 100000000
   2. 000000 01000 00000
4. 0x400
   1. 1024, 10000000000
   2. 000001 00000 00000
5. 0x1049
   1. 4169, 1000001001001
   2. 000100 00010 01001

Define the words "pointer" and "memory address".

Pointer is a program pointing out the memory address.

Memory address is a section of memory which has own unique number of address.

Demonstrate that you have a working C compiler setup where you will be able to write programs for class. Capture the output of the compilation and execution of a simple program (such as "Hello, world").



This week we have talked about system calls. Take the following short program and produce the assembler output from the compiler. Include the assembler file in your homework.

|  |
| --- |
| #include <unistd.h>  int main() {  char \*buf = "123";  write(1, buf, 3);  return 0;  } |

* 1. Identify the instruction that calls the string output function. Is it a library function or a system call? How can you tell?
     1. A. library function. Reason: If I remove “#include<unistd.h>”, it returns compile error.
  2. Identify the arguments to the function. How many are there?
     1. No arguments to the main function.
     2. Three arguments to the Write function(1(int), buf(char), 3(int))
  3. If your string output function is a library function rather than system call, can you find the function and instruction that actually does the trap into the kernel?
     1. Inside header file?

|  |
| --- |
| #include <stdio.h>  #include <time.h>  int main(int argc, char \*args[]){  clock\_t start, end;  start = clock();  printf("Hello world!\n");  end = clock();  printf("It takes %f sec to execute.\n", (double)(end-start)/CLOCKS\_PER\_SEC);  return 0;  } |

1. Identify the instruction that calls the string output function. Is it a library function or a system call? How can you tell?\_
   1. library function. Reason: If I remove “#include<unistd.h>”, it returns compile error.
2. Identify the arguments to the function. How many are there?
   1. Two arguments to the main function.
   2. One argument to the first printf function and two arguments to the second printf function.
3. If your string output function is a library function rather than system call, can you find the function and instruction that actually does the trap into the kernel?
   1. Inside the header files?