APS105: Lecture 2 Notes

Last lecture: Introduction to course

Announcements:

1) New instructor for LEC 0104, Salma Emara
2) Office hours: Mondays 12:30pm - 1:30pm
   Zoom link can be found in the syllabus
3) Lab0 is due Jan 15 11:59pm
   Lab1 is due Jan 22 11:59pm

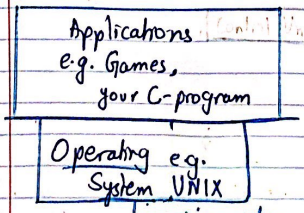
11:15

Today's lecture: ① Introduce Basic Computer Structure
Objectives ② Write a simple C Program taking input and printing output

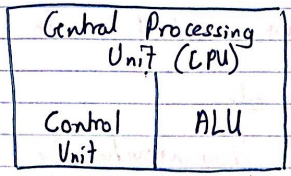
To give a context on how your C-program fits in the computer system, let's introduce basic structure of computers.

- A sequence of instructions executed or performed by the computer

Software



Hardware



→ executes instructions in native/machine code
e.g. adds two numbers

Main memory → stores data and instructions that can be sent to the CPU to execute

I/O (Input/Output) Devices → connects the computer system to the user/world
e.g. input: keyboard
output: monitor

11:25 How is data and instructions stored?

Data & Numbers are encoded in binary:

Smallest unit: a bit (either 0 or 1)
Byte (B): 8 bits
KiloByte (KB): 1024 Bytes
MegaByte (MB): 1024 KB
GigaByte (GB): 1024 MB
TeraByte (TB): 1024 GB

For example, a number is represented as binary

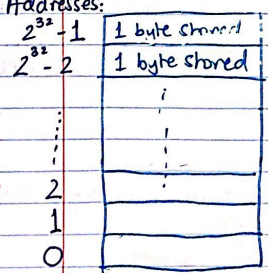
0	0000	0000
1	0000	0001
2	0000	0010
3	0000	0011
4	0000	0100
5	0000	0101
6	0000	0110
7	0000	0111
8	0000	1000

To represent 0-3 → require 2 bits (4 numbers = $2^{2\text{bits}}$)

To represent 0-7 → require 3 bits (8 numbers = $2^{3\text{bits}}$)

To represent 0-? → require 8 bits ($2^8 - 1 = 255$; ? numbers = $2^{8\text{bits}} = 256$)

Back to the main memory, it is organized to be able to access the data/instructions there. It is divided into cells. Each cell holds 1 byte. Each cell has an address, hence the memory is "byte-addressable". Addresses too are represented in binary.



Ex: If I choose to represent addresses using 32-bits, how many different addresses can I have?

$= 2^{32}$ addresses, each address accesses 1 byte

Total memory that can be stored $2^{32} * 1$ byte

$2^{32} * 1\ \text{byte} = 2^2 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10}$

kB

MB

GB

4 GB of main memory

In modern computers, addresses are 64-bit.

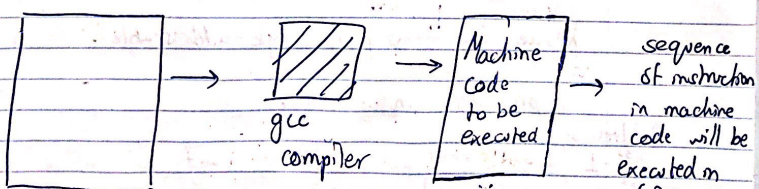
Homework: how big can my memory be, if I have a 64-bit address?

11:25

Note: The unit is not Giga or Tera bytes (Bigger!)

Back to C-program, to develop a C-code

① Edit ② Compile ③ Run



lab1.c

11:30

Note: Machine code is stored in main memory, but it is executed on the CPU. (performed)

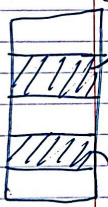
11:30 Write a C-program that takes from the user the number of pizzas, and outputs how many slices we have in total.

Assumptions: 1 pizza = 8 slices

```
#include <stdio.h>   → standard library to use input & output functions of printf and scanf.

int main (void) {   → every C-program must have a main function, where the program starts
```

Memory




```
int input;
int output;
```
→ declares two numbers, "variables", and reserve a space for them in main memory.

int: number with no decimals.

every instruction ends with ;

```
printf("Enter number of pizzas:");
```
↓ displays text on screen

```
scanf("%d", &input);
```
→ go to input variable in main memory, and put the number typed there

%d: format specifier, expecting an int

```
output = input * 8;
```
→ instruction/command storing input * 8 into output in memory

```
printf("We have %d slices from %d pizzas \n", input, output);
```
%d will print value of input; %d will print value of output; \n → newline

```
return 0;
```
→ stops program!

} → closes opening bracket of main function

Show Demo on VS Code!

- Better naming of variables (① Camel Case e.g. inputPizza, outputSlices ② Snake case input_pizza, output_slices)
- Indentation
- How to write coments
- Escape sequences

To print "

```
printf( " \" " );
```

backslash is to avoid confusion with " of printf and what is to be printed

To print \

```
printf ( " \\ " ) ;
```

backslash to avoid confusion with \ in \n

Heads-up on next lecture:

- int v.s. double

5 int : stores a number without any decimal points or numbers after decimal

0 double : stores a number with decimal point and decimal places, "floating point number"