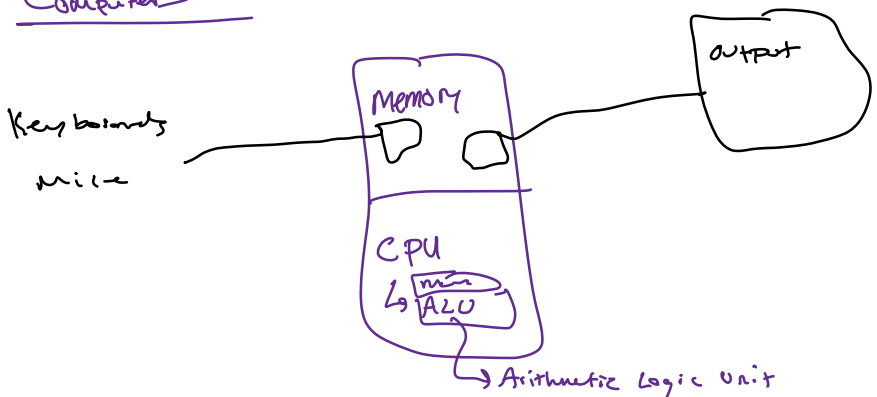
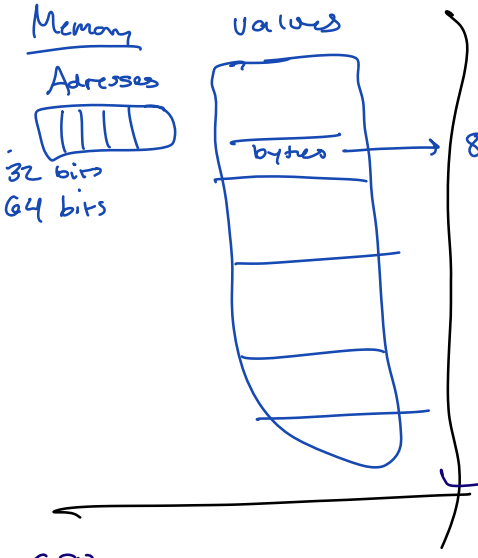
Computers



Arithmetic Logic Unit
processes data from memory using instructions from memory

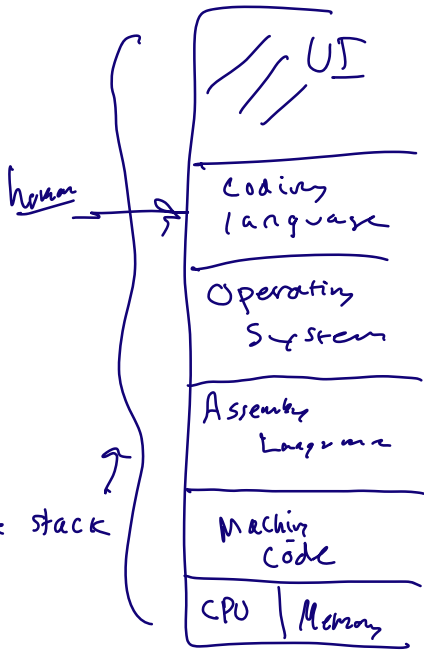


8 bits = 1 byte = bits → {0, 1}

1024 bytes Kb
1M. bytes Mb
billion Gb

CPU
takes instructions
takes memory address
and does

ex) ADD — Adress 1 — Adress 2 — store the result

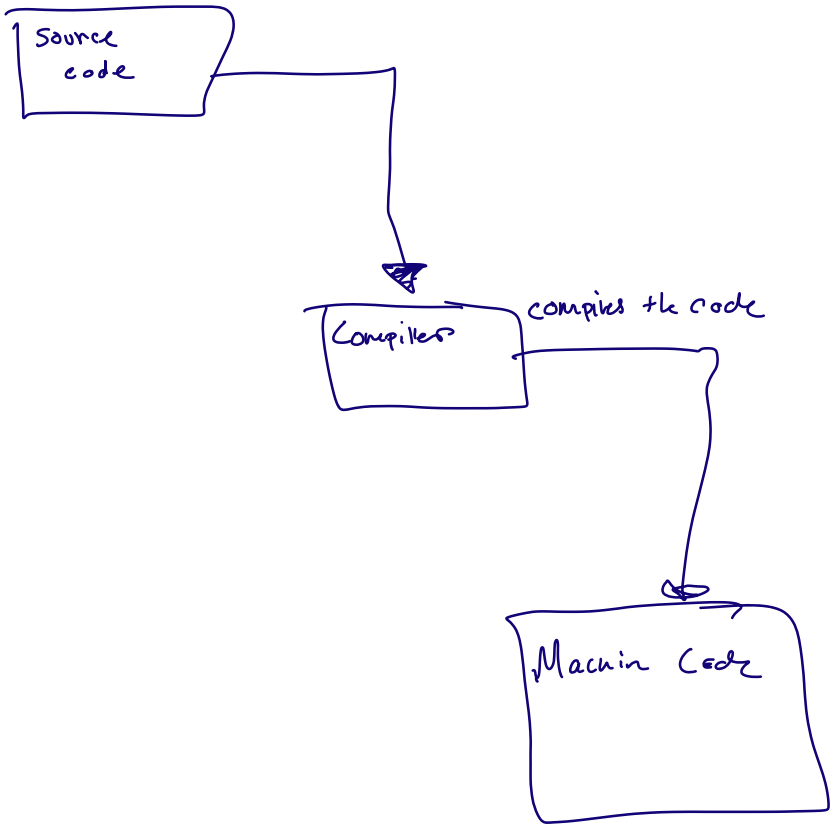


C → compilled

Java →

C++, C# →

Swift →

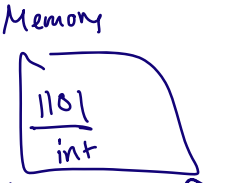


Scripting → have an interpreator (which compiled)

- Python
- Bash (shell scripted)
- Calculator input
- f

Type + Data = meaning

↳ a specification for data (bits) interpretation



basic (primitive)

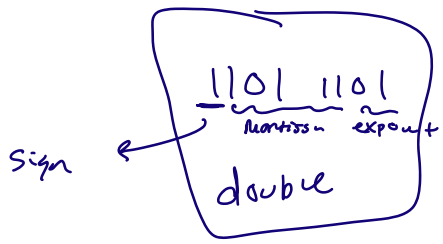
- integer (int) -32767 to 32767
- doubles (double)
- Float (float)
- boolean (bool)
- unsigned interger (unsigned int)
- long (long)
- Characters (char) 'c'

$13 = 2^3 + 2^2 + 2^0$

base 10 integer

→ Sign, exponent, mantisia

Sign • mantissa × $2^{exponent}$



Variable → a typed spot in memory where can store things

① declare → name a spot and give its type : type vname

② Assign → =

variable = expression ← assignment statement

example:

```
int myInt; //declares a variable myInt with type int
myInt = 0;
myInt = 1;
char myFirstInitial = 'A';
```

Operators → a symbol for performing a function

+ → plus ex) int int2 = 6;
int sum = 0;
sum = sum + 3;
sum = sum + int2;
↳ infix

* → Mutip.

/ → Division

% → modus → 3%2 = 1 $\frac{3}{2}$ has a remainder of 1

- → Subtract ex) int2 = sum - 3;
int2 = - sum;

Naming

can't start with

printf → printf (string, variables or numbers) ← print list

printf("hello\n"); → hello

int int2 = 6;
printf("hello: the value int2 is %d", int2);

printf("value 1 is %d and value2 is %d", 6, int2);

scanf → to get input from user

double miles;
scanf("%lf", &miles);

%c → character (char)

%d int

%f double → printing out double in printf

%lf → reading in double in scanf

printf / scanf
printf (scanf

```
int myInt;
scanf("%d", &myInt);
```
↳ address

Comments

// ← comment sing

/*
⋮ } multilike comments
*/

Mixed Types Expressions

For example

```
int myInt 2;
double myDouble 2.0;
```

2 + 2.0 ?

↓

int or double → double

```
int product;
product = (int)(myInt + myDouble);
```
↖ type cast

Example

```
int numStudents;
int total_score;
double average;
average = (double)total_score / (double)num_students;
```
→ This does integer division
3 2

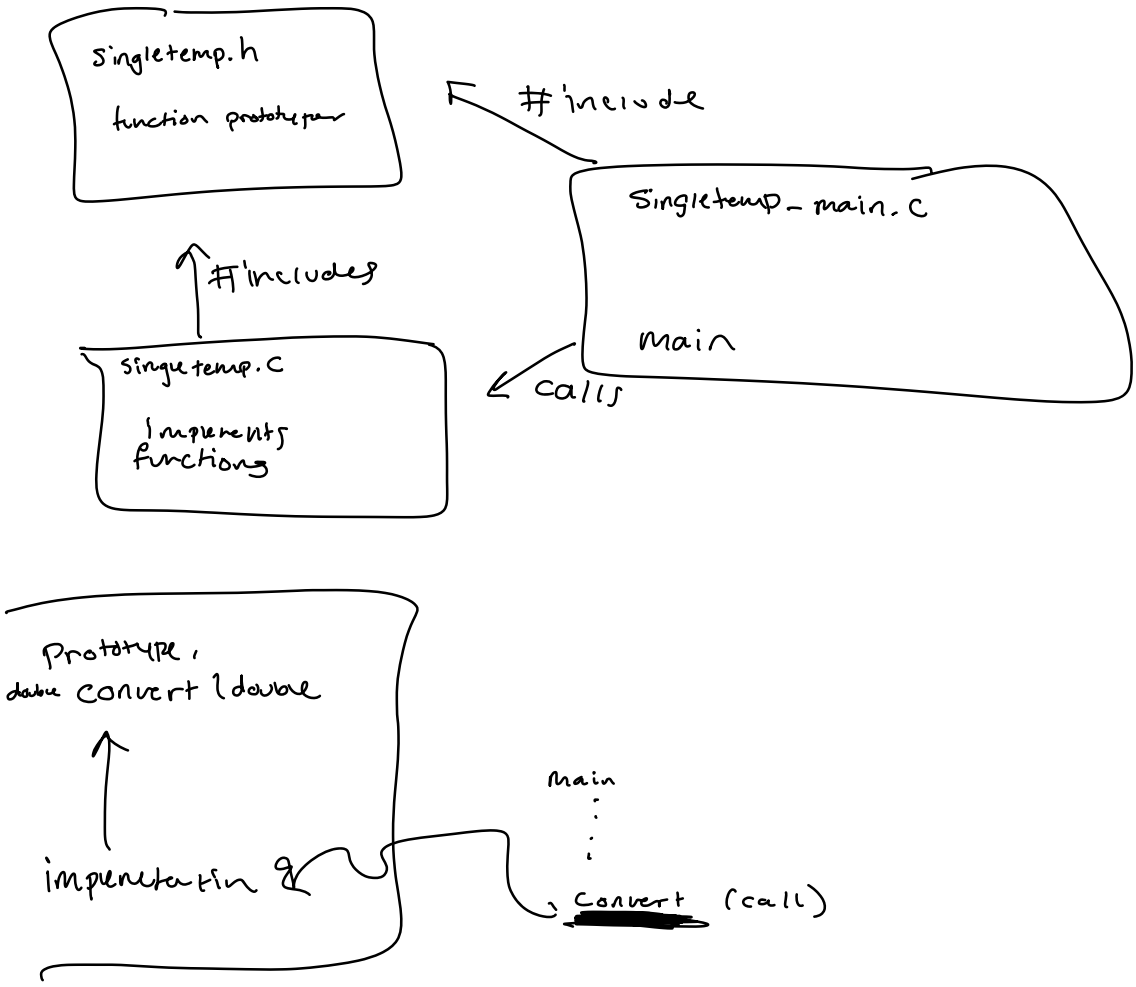
Rounding

```
double x;
int rounded_x;
rounded = (int)(x + 0.5)
```

7.73.

↓

7


singletemp.h
function prototypes
#include
singletemp_main.c
main
#includes
singletemp.c
implements functions
calls
Prototype:
double convert (double)
implementation
main
convert (call)