APS105: Lecture 3

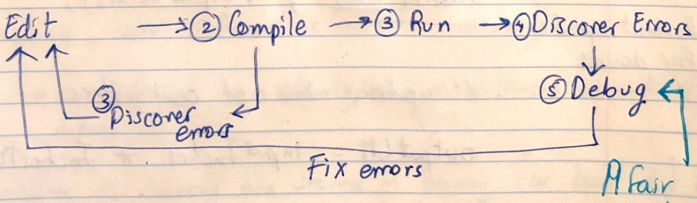
Jan 14, 2022

Last lecture: Basic structure of Computers
Example program with variables, printf, scanf

Today: Discuss constant variables, data types and data representation

Important Note: The proper way of learning programming is to go through the development cycle to understand/visualize how it works & if it doesn't work → learn how to fix it

Recap on development cycle



A fair amount of time is spent here!

Don't start writing code before having a plan/outline → it can be a diagram of how your code should function

11:45 – DEMO on VS Code –

Example Program: Convert Inches to Centimetres

```
//Description: This program converts Inches to Centimetres
#include <stdio.h>

int main (void) {
    // Variable Declaration
    const double InchesToCM = 2.54;

    double inputInches, outputCM;

    //Prompt user to enter # inches
    printf("Enter number of inches to convert: ");

    scanf("%lf", &inputInches);

    //Compute number of centimetres

    outputCM = inputInches * InchesToCM;

    //Display result
    printf("The number of cm is %5.2lf.\n",
           outputCM);

    return 0;
}
```

Keyword specifying variable is fixed & cannot be changed in the code

variable data type that allows storing a number with decimal points

use camel case throughout the course for const, also capitalize first letter

format specifier for double

only rounds to display, but doesn't round actual variable in memory.

lf: double
.2: round to 2 decimal places
5: leave 5 spaces before printing double

- So far we've used variables, stored in main memory, that contain numbers

Integers
- 0
- +ve whole #
- -ve whole #

fractional or real numbers
- these contain a decimal number

- As you make a rough plan of your software, you need to decide on your variable types.

e.g. if you have a variable storing # of attendees - int
if you have a variable storing interest rate - double
if you have a variable storing number of cm in inch - const double

- Variables have different sizes in memory
  - ↳ ie. different variable types use different amounts of memory.
- variables in memory are stored in binary (or base 2) in bits

Recap:

1 bit: either 0 or 1 → 2 numbers
2 bits: 00, 01, 10, 11 → 4 numbers
3 bits: $b_2 b_1 b_0$

0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

→ 8 numbers

To convert from binary to decimal:

e.g. 101

equation: $b_0 * 2^0 + b_1 * 2^1 + b_2 * 2^2$
$1 * 1 + 0 * 2 + 1 * 4 = 5$

or easily, 4 2 1
1 0 1 = 5

4 bits: 8 4 2 1
1101 = 8 + 4 + 0 + 1 = 13
→ largest # is 1111 = 8 + 4 + 2 + 1 = 15
$2^4 - 1$

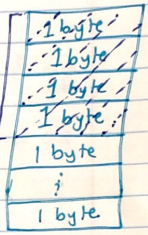
n bit: largest # is $2^n - 1$
and we can represent 2^n numbers using n bits

So what data types do we have, and how are they stored in memory:

I - int

```
int numberOfAttendees;
```

reserve →



* reserves 32 bits of memory, i.e.

$\frac{32 \text{ bits}}{8 \text{ bits/byte}} = 4$ bytes

* the number can be +ve or -ve
  ⇒ signed

  1 bit for sign
  31 bits for representing the #

* maximum range $-2^{31} \ldots\ldots 0 \ldots\ldots 2^{31}-1$

  2^{31} numbers 2^{31} numbers

  total numbers = $2^{31} + 2^{31}$
  $= 2^{31} \cdot 2^1 = 2^{32}$ numbers

other types of ints:

unsigned int : $0 \rightarrow 2^{32}-1$
short : 16 bits - 2 Bytes
long : 32 bits - 4 Bytes
long long : 64 bits - 8 Bytes

You will only be questioned on int data type

format specifier is %d

II - Floating Point or Real Numbers

```
double inputInches;
```

represents real numbers

floating point binary is similar to standard scientific notation:

2.99×10^{8} or 2.99e8 or 2.99E8

(2.99 called mantissa; 8 called exponent)

The floating point binary stores mantissa & exponent separately

double → 64 bits → "double precision"

float → 32 bits → "single precision"

Homework: how many bits do each have for sign, mantissa, exponent?

We only use double in this course

↳ format specifier is %lf

III - Char type

```
char firstNameInitial = 'S';
```

represents a single letter, number digit, symbol

A - Z , a - z , 0 - 9 , $, @, ?, ,

stored in 1 byte - 8 bits - 2^8 possible characters

format specifier of char

printf("First Name Initial %c", firstNameInitial);

ASCII codes are used to represent character, no need to know about them

'0' = has an ASCII code of 48

in memory it is stored as

128	64	32	16	8	4	2	1
0	0	1	1	0	0	0	0

IV - Boolean Type

Need to include standard bool library

```
#include <stdbool.h>

int main (void) {

    bool covid = false;

}
```

represents two values: true or false

uses 1 byte of memory