Types

Last updated 8/20/20

Statements

- These slides introduce C types

- Upon completion: You should be able to interpret and code using types

Types

Quick Aside:

- A variable is a name for some entity stored in memory

- We refer to the entity by name (the variable) because we don't know the value

- Just like in algebra
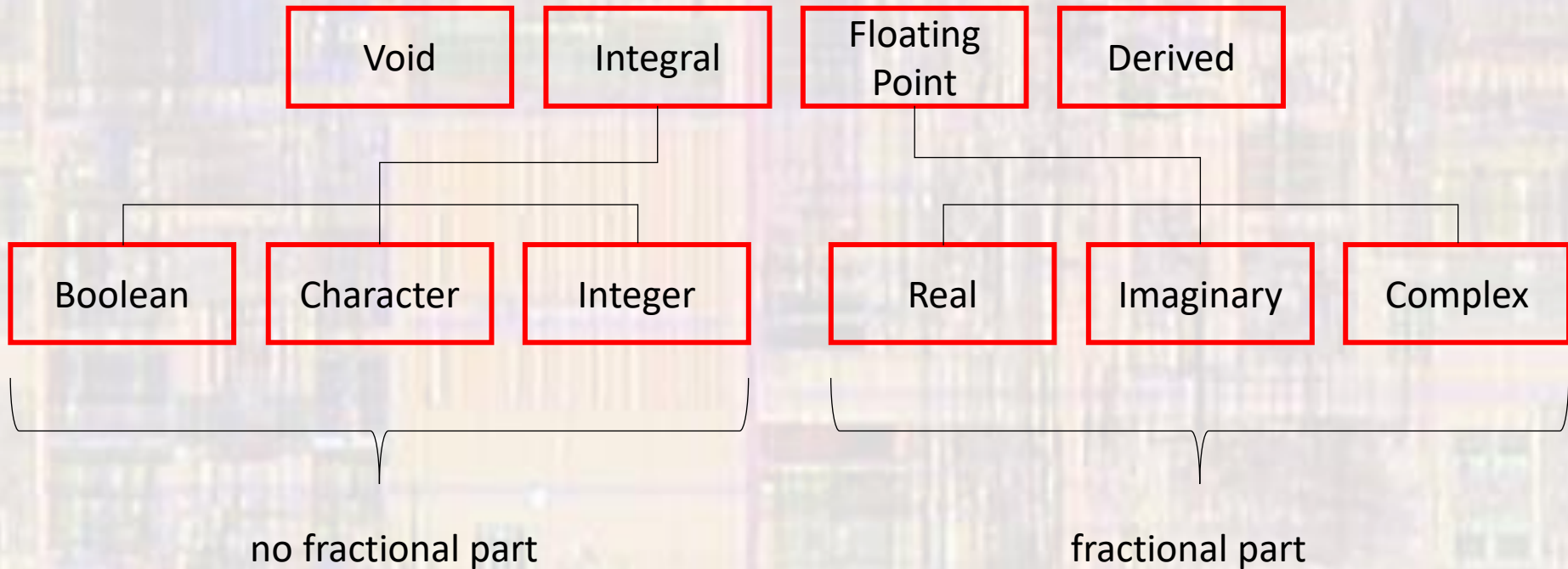
Types

- What is a Type

  - The "space" in which a variable is defined

  - Space
    - All possible allowed values
    - All defined operations

  - Integer Space
    - whole numbers
    - +, -, x
    - No divide

Types

- Why Types

  - No room for confusion in a computer
    - Must get the same answer every time

  - Everything must be stored into memory somewhere
    - Program Memory
    - Data Memory

  - Memory used to be expensive
    - Minimize the amount needed

Types

- C Types

```
┌──────────┐  ┌──────────┐      ┌──────────┐  ┌──────────┐
│   Void   │  │ Integral │      │ Floating │  │ Derived  │
│          │  │          │      │  Point   │  │          │
└──────────┘  └──────────┘      └──────────┘  └──────────┘
```

| Boolean | Character | Integer | | Real | Imaginary | Complex |

no fractional part   fractional part

Types

- C Types - void

  - No values
  - No defined operations

  - Used when we want to indicate that nothing is here

  - Examples

    MyFunction(void);
    // call a function with no input parameters

    void YourFunction(int val){   …
    // indicate that a function returns nothing

Types

- C Types - bool - boolean

  - 2 values
    - true, false

  - Logical operations
    - and (&&), or (||), not (!)

  - Examples
    ```
    bool in1;            // declare 3 variables of type bool
    bool in2;
    bool in3;

    …
    in1 && in2;
    // in1 AND in2


    in3 = !(in1 || in2);
    // in3 = not(in1 OR in2) = in1 NOR in2
    ```

Types

- C Types – char - character

    - ASCII - 128 values
        - a,b,c,1,2,3,$,%,*, …
        - English language characters
    - Unicode – millions of values

    - Stored in the computer as integers
    - Same operations as integers
    - Become characters when visualized
    - Require a single quote

    - Examples
        char initial1 = 't';
        char initial2 = 'j';

        printf("%c%c", initial1, initial2);
        // print - tj

        printf("%c", (initial1 - initial2));
        // print – (line feed)
        // 116 – 106 = 10 → linefeed

ASCII TABLE

NULL – no character

numbers add 0x30

upper and lower case differ by hex 0x20

Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]
1	1	1	1	[START OF HEADING]
2	2	10	2	[START OF TEXT]
3	3	11	3	[END OF TEXT]
4	4	100	4	[END OF TRANSMISSION]
5	5	101	5	[ENQUIRY]
6	6	110	6	[ACKNOWLEDGE]
7	7	111	7	[BELL]
8	8	1000	10	[BACKSPACE]
9	9	1001	11	[HORIZONTAL TAB]
10	A	1010	12	[LINE FEED]
11	B	1011	13	[VERTICAL TAB]
12	C	1100	14	[FORM FEED]
13	D	1101	15	[CARRIAGE RETURN]
14	E	1110	16	[SHIFT OUT]
15	F	1111	17	[SHIFT IN]
16	10	10000	20	[DATA LINK ESCAPE]
17	11	10001	21	[DEVICE CONTROL 1]
18	12	10010	22	[DEVICE CONTROL 2]
19	13	10011	23	[DEVICE CONTROL 3]
20	14	10100	24	[DEVICE CONTROL 4]
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]
22	16	10110	26	[SYNCHRONOUS IDLE]
23	17	10111	27	[ENG OF TRANS. BLOCK]
24	18	11000	30	[CANCEL]
25	19	11001	31	[END OF MEDIUM]
26	1A	11010	32	[SUBSTITUTE]
27	1B	11011	33	[ESCAPE]
28	1C	11100	34	[FILE SEPARATOR]
29	1D	11101	35	[GROUP SEPARATOR]
30	1E	11110	36	[RECORD SEPARATOR]
31	1F	11111	37	[UNIT SEPARATOR]
32	20	100000	40	[SPACE]
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/

Decimal	Hexadecimal	Binary	Octal	Char
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F
71	47	1000111	107	G
72	48	1001000	110	H
73	49	1001001	111	I
74	4A	1001010	112	J
75	4B	1001011	113	K
76	4C	1001100	114	L
77	4D	1001101	115	M
78	4E	1001110	116	N
79	4F	1001111	117	O
80	50	1010000	120	P
81	51	1010001	121	Q
82	52	1010010	122	R
83	53	1010011	123	S
84	54	1010100	124	T
85	55	1010101	125	U
86	56	1010110	126	V
87	57	1010111	127	W
88	58	1011000	130	X
89	59	1011001	131	Y
90	5A	1011010	132	Z
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_

| Decimal	Hexadecimal	Binary	Octal	Char
96	60	1100000	140	`
97	61	1100001	141	a
98	62	1100010	142	b
99	63	1100011	143	c
100	64	1100100	144	d
101	65	1100101	145	e
102	66	1100110	146	f
103	67	1100111	147	g
104	68	1101000	150	h
105	69	1101001	151	i
106	6A	1101010	152	j
107	6B	1101011	153	k
108	6C	1101100	154	l
109	6D	1101101	155	m
110	6E	1101110	156	n
111	6F	1101111	157	o
112	70	1110000	160	p
113	71	1110001	161	q
114	72	1110010	162	r
115	73	1110011	163	s
116	74	1110100	164	t
117	75	1110101	165	u
118	76	1110110	166	v
119	77	1110111	167	w
120	78	1111000	170	x
121	79	1111001	171	y
122	7A	1111010	172	z
123	7B	1111011	173	{
124	7C	1111100	174		
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	[DEL]

Types

- C Types – int - integer

  - Values are system dependent
    - integers only
    - 2, 4, 8 bytes
    - short int, int, long int, long long int

  - Operations
    - Arithmetic operations   +, -, *, /, %
    - Comparison operations   <, >, <=, >=, ==, !=
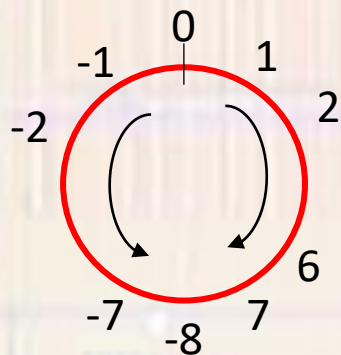   - Bitwise operations   ~, |, &, ^, <<, >>

  - Examples
    ```
    int aa;                  // declare 3 variables of type int
    int bb;
    int cc;
    …
    aa + bb;
    // add aa to bb
    ```
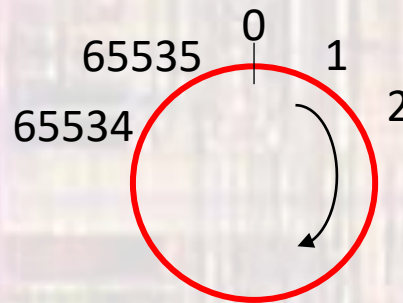
Types

- C Types – int - integer

  - Special considerations with type int
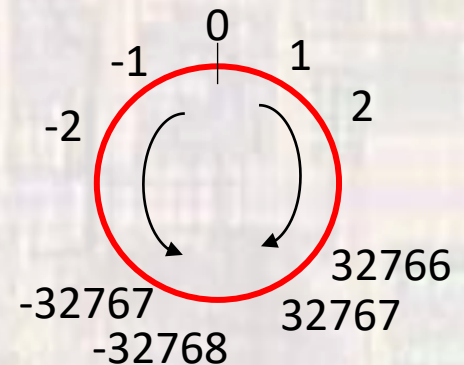   - Range is defined and limited
    - SIGNED and UNSIGNED variants
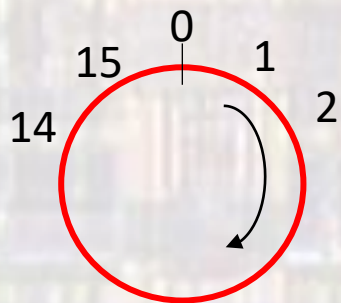
4 bit unsigned   4 bit signed   16 bit unsigned   16 bit signed

Types

- C Types – int - integer

  - Consider 3 16bit variables of type short int

    aa = 32,500   bb = 300   cc = 15,000

	signed	unsigned
aa + bb =	-32,736	32,800
aa/cc =	2	

Types

- C Types – special integers

  - A special set of integers are defined for embedded systems
    - Designed to allow register/memory access
    - Not system dependent

  ```
  #include <stdint.h>
  ```

signed char	**int8_t;**
unsigned char	**uint8_t;**
short	**int16_t;**
unsigned short	**uint16_t;**
int	**int32_t;**
unsigned	**uint32_t;**
long long	**int64_t;**
unsigned long long	**uint64_t;**

Types

- C Types – int - integer

  - TI –MSP432

  ```
  COM6

  üsize of short int = 2
  example short int: 32767
  example short int + 1: -32768

  size of plain int = 4
  example plain int: 2147483647
  example plain int + 1: -2147483648

  size of long int = 4
  example long int: 2147483647
  example long int + 1: -2147483648
  ```

  - Laptop

  ```
  size of short int = 2
  example short int: 32767
  example short int + 1: -32768

  size of plain int = 4
  example plain int: 2147483647
  example plain int + 1: -2147483648

  size of long int = 4
  example long int: 2147483647
  example long int + 1: -2147483648

  size of long long int = 8
  example long long int: -1
  example long long int + 1: 0
  ```

Types

- C Types – float – real

  - Values are system dependent
    - SIGNED
    - 4 byte – 1,8,23
    - 8 byte – 1,11,52
    - float, double, long double

  - Operations
    - Arithmetic operations   +, -, *, /
    - Comparison operations   <, >, <=, >=, ==, !=

  - Examples
    float aa;   // declare 3 variables of type float
    float bb;
    float cc;
    …
    aa / bb;
    // aa divided by bb

Types

- C Types – float – imaginary

  - Values are system dependent
    - SIGNED
    - 4 byte – 1,8,23
    - 8 byte – 1,11,52
    - float imaginary, double imaginary, long double imaginary

  - Operations
    - Arithmetic operations   +, -, *, /
    - Comparison operations   <, >, <=, >=, ==, !=

  - Not supported in many systems

Types

- C Types – float – complex

  - Values are system dependent
    - SIGNED
    - 4 byte – 1,8,23
    - 8 byte – 1,11,52
    - float complex, double complex, long double complex

  - Operations
    - Arithmetic operations   +, -, *, /
    - Comparison operations   <, >, <=, >=, ==, !=

  - Real and imaginary parts must be the same size

Types

- C Types – float - real

  - Consider 3 variables of type float
    aa = 2.5   bb = 300.5   cc = 0.035

    aa + bb =   303.0

    aa/cc =   71.428571

Types

- Special Details
  - Functions:

    sizeof(type)

    sizeof expression

    typeof(expression)

  - Include

    \<limits.h\>

    \<float.h\>

    \<stdint.h\>

SHRT_MIN	Minimum value for an object of type short int
SHRT_MAX	Maximum value for an object of type short int
USHRT_MAX	Maximum value for an object of type unsigned short int
INT_MIN	Minimum value for an object of type int
INT_MAX	Maximum value for an object of type int

    defines max and min values for standard types