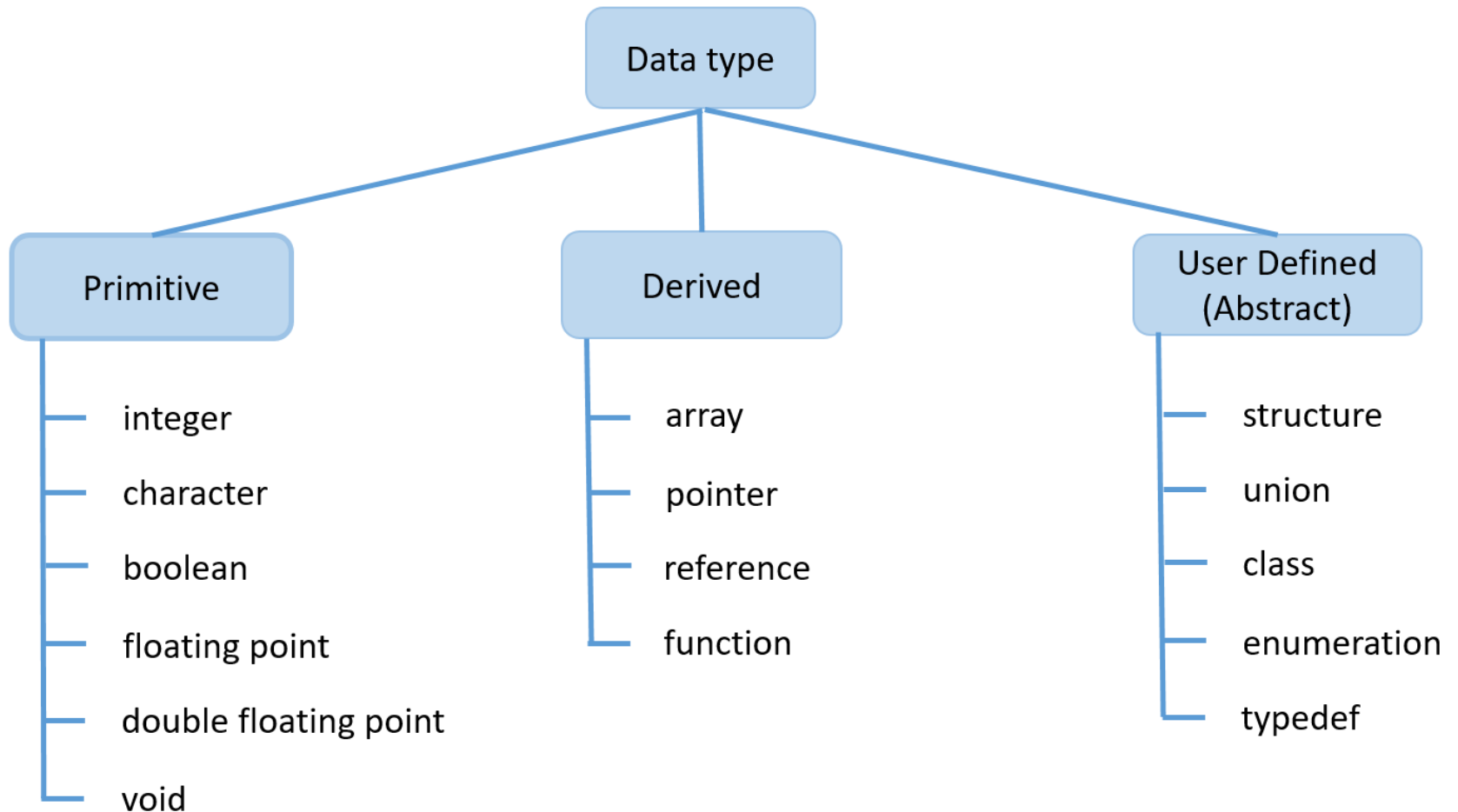


บทที่ 1

โครงสร้างข้อมูลเบื้องต้น

สุนทรี คุ่มไพโรจน์

Data Type ในภาษา C



ที่มา: <https://www.geeksforgeeks.org/c-data-types/>

Basic Data Type

ชนิดข้อมูล	Memory (Bytes)	Range(พิสัย) <i>ในหน่วย 2</i>	Format in C
char	1	-128 ถึง 127	%c
signed char	1	-128 ถึง 127	%c
unsigned char	1	0 ถึง 255	%u
int	4	-2,147,483,648 ถึง 2,147,483,647	%d
unsigned int	4	0 ถึง 4,294,967,295	%u
short int	2	-32,768 ถึง 32,767	%hd
unsigned short	2	0 ถึง 65,535	%hu
long	8	$-(2^{63})$ to $(2^{63})-1$	%ld
float	4		%f
double	8		%lf

<https://www.geeksforgeeks.org/data-types-in-c/>

พิสัยในการเก็บข้อมูล

- พิสัย(ช่วงข้อมูล)ในการเก็บข้อมูลที่เป็นเลขฐานสอง (Binary system) นั้นคำนวณจากจำนวนบิตที่ใช้ในการจัดเก็บต่อหนึ่งตัวอักษรหรือตัวเลขหนึ่ง
 - 1 bytes = 8 bits
 - กรณีของ `int` ใช้ 4 bytes = 32 bits
บิตแรกแทนเครื่องหมาย
เหลือจำนวนบิตที่แปลงเป็นตัวเลขได้ 31 บิต
 $2^{31} = 2,147,483,648$
ดังนั้นหากจัดเก็บในรูป 2's complement
พิสัยของ int คือ - 2,147,483,648 ถึง 2,147,483,647
- ตัวอย่างโปรแกรมพิมพ์ค่าสูงสุดต่ำสุดของจำนวนเต็มชนิดต่างๆ(L1.C)

L1.c

```
1 // Program MaxMin.c
2
3 #include <stdio.h>
4 #include <limits.h>
5
6 int main() {
7     printf("Constants\t Value\n");
8     printf("CHAR_BIT\t %d\n", CHAR_BIT);
9     printf("CHAR_MAX\t %d\n", CHAR_MAX);
10    printf("CHAR_MIN\t %d\n", CHAR_MIN);
11    printf("SCHAR_MAX\t %d\n", SCHAR_MAX);
12    printf("SCHAR_MIN\t %d\n", SCHAR_MIN);
13    printf("UCHAR_MAX\t %u\n", UCHAR_MAX);
14    printf("INT_MAX\t %d\n", INT_MAX);
15    printf("INT_MIN\t %d\n", INT_MIN);
16    printf("UINT_MAX\t %u\n", UINT_MAX);
17    printf("SHRT_MAX\t %hd\n", SHRT_MAX);
18    printf("SHRT_MIN\t %hd\n", SHRT_MIN);
19    printf("USHRT_MAX\t %hu\n", USHRT_MAX);
20    printf("LONG_MAX\t %ld\n", LONG_MAX);
21    printf("LONG_MIN\t %ld\n", LONG_MIN);
22    printf("ULONG_MAX\t %lu\n", ULONG_MAX);
23    return 0;
24 }
```

เลขจำนวนเต็ม (Integer)

- เซตของจำนวนเต็ม
 $\{ \dots, -3, -2, -1, 0, 1, 2, 3, \dots \}$
- เก็บข้อมูลในหน่วยความจำ(memory)ของคอมพิวเตอร์
ในรูปเลขฐานสอง
ตัวอย่าง integer ที่มีค่าเท่ากับ 13 สามารถเก็บได้ดังนี้

0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

16

ข้อมูลชนิดจำนวนเต็มในภาษา C

- ตัวอย่างการประกาศข้อมูลชนิดจำนวนเต็ม

```
int age;
```

- ตัวอย่างการกำหนดค่าให้กับตัวแปร

```
age = 15;
```

- ตัวอย่างโปรแกรมพิมพ์พิกัดของจุด(L2.c)

L2.c

```
1 // Program: Coord.c
2 // จากหนังสือเรียน สสวท.
3 #include <stdio.h>
4 void main() {
5     int x1 , y1;
6     int x2=5, y2=0;
7     x1=2;
8     y1=3;
9     printf("The first coordinate is (%d, %d).\n", x1,y1);
10    printf("The second coordinate is (%d, %d).\n", x2,y2);
11 }
12
```

ที่มา [1] การเขียนโปรแกรมคอมพิวเตอร์ภาษาซี. ผศ.อุมพร ศิริธรรานนท์, ผศ.กัลยาณี บรรจงจิตร,
รศ.ดร.นวลวรรณ สุนทรภิชช์. โครงการตำราวิทยาศาสตร์และคณิตศาสตร์ มวลนิธิ สอน. 2006.

เลขฐาน

- คอมพิวเตอร์มีพื้นฐานมาจากเลขฐาน 2
- ตัวอย่างเลขฐาน 2 $(1101)_2 = (15)_8$
$$= 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$
$$= 8 + 4 + 0 + 1$$
$$= (13)_{10} = 1 \times 10^1 + 3 \times 10^0$$
- ตัวอย่างโปรแกรมพิมพ์เลขฐานต่างๆ (L3.C)

L3.c

```
1 // program: prnD-O-X.c
2 #include <stdio.h>
3
4 void main() {
5     int a=78;
6
7     printf("Value %d in decimal is %d.\n", a,a);
8     printf("Value %d in octal is %o.\n", a,a);
9     printf("Value %d in hexadecimal is ", a);
10    printf("%x  or %X. \n", a,a);
11
12 }
13
```

แปลงเลขฐาน(ทบทวน)

- จาก ฐาน 10 \rightarrow 53 ไปเป็น ฐาน 2
- จาก ฐาน 10 \rightarrow 53 ไปเป็น ฐาน 8
- จาก ฐาน 10 \rightarrow 53 ไปเป็น ฐาน 16
- จาก ฐาน 2 \rightarrow 11101 ไปเป็น ฐาน 10
- จาก ฐาน 8 \rightarrow 57 ไปเป็น ฐาน 10
- จาก ฐาน 16 \rightarrow 5A ไปเป็น ฐาน 10
- จาก ฐาน 10 \rightarrow 8 ไปเป็น ฐาน 3
- นาฬิกา ก็ใช้ระบบเลขฐาน เหมือนกัน ?

Binary number representation

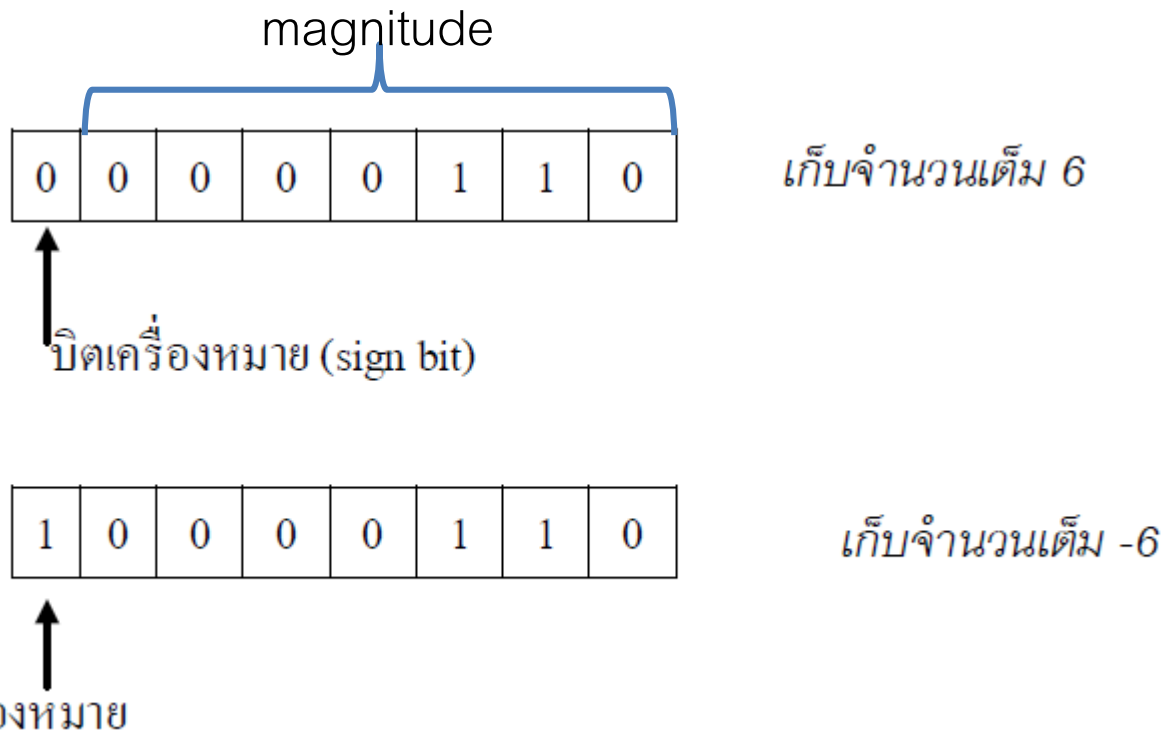
การแทนเลขจำนวนเต็มด้วยเลขฐาน 2

- Unsigned representation ไม่ใส่เครื่องหมาย
- Signed magnitude representation มีเครื่องหมาย
- 1's complement system ใส่เครื่องหมาย.
- 2's Complement System " ——— "

Binary value	Unsigned interpretation	Signed magnitude representation	Ones' complement interpretation	Twos' complement interpretation
00000000	0	+0	+0	+0
00000001	1	1	1	1
⋮	⋮	⋮	⋮	⋮
01111101	125	125	125	125
01111110	126	126	126	126
01111111	127	127	127	127
10000000	128	-0	-127	-128
10000001	129	-1	-126	-127
10000010	130	-2	-125	-126
⋮	⋮	⋮	⋮	⋮
11111101	253	-125	-2	-3
11111110	254	-126	-1	-2
11111111	255	-127	-0	-1

Signed magnitude representation

- มีการแบ่งพื้นที่ที่เป็น 2 ส่วน คือส่วน magnitude (ค่าขนาดของตัวเลข) และ sign bit ที่จะอยู่ที่บิตที่มีค่าสูงสุด
- ตัวอย่างการเก็บ จำนวนเต็ม 6 และ -6



ที่มา: บิตเครื่องหมาย

<http://www.cs.gordon.edu/courses/cps111/Notes/Binary%20number%20system/binary.html>

1's complement system(ส่วนเติมเต็มหนึ่ง)

- สำหรับเลขจำนวนเต็มบวก ใช้ตัวเลข **binary** ที่คำนวณมาได้เลย
- สำหรับเลขจำนวนเต็มลบ นำตัวเลข **binary** ที่คำนวณได้มาเปลี่ยนค่าสถานะของทุกบิต (bit) ให้เป็นบิตตรงกันข้าม
(ถ้าบิตเป็น 1 ให้เปลี่ยนเป็น 0 และถ้าบิตเป็น 0 ให้เปลี่ยนเป็น 1)
- ตัวอย่างที่ 1
$$+12 = 0000\ 1100$$
$$-12 = 1111\ 0011 \text{ (ส่วนเติมเต็มหนึ่ง)}$$
- ตัวอย่างที่ 2
$$+18 = 0001\ 0010$$
$$-18 = 1110\ 1101 \text{ (ส่วนเติมเต็มหนึ่ง)}$$

2's Complement System(ส่วนเติมเต็มสอง)

- สำหรับเลขจำนวนเต็มบวก ใช้ตัวเลข **binary** ที่คำนวณมาได้เลย
- สำหรับเลขจำนวนเต็มลบ ให้นำตัวเลข 1's complement บวก 1 ที่บิตขวาสุด(หลังสุด)
- ตัวอย่างที่ 1
- $+12 = 0000\ 1100$
- $-12 = 1111\ 0011$ (ส่วนเติมเต็มหนึ่ง)
- $1111\ 0011 + 1$
- $-12 = 1111\ 0100$ (ส่วนเติมเต็มสอง)

ที่มา: <https://www.geeksforgeeks.org/binary-representations-in-digital-logic/>

การดำเนินการบวกเลขฐาน

- ให้นำเลขฐานสองของทั้ง 2 จำนวนมาบวกกัน
- ตัวอย่าง การหาผลบวกของ $5 + 7$
- 00000101 คือ 5
- $+ \underline{00000111}$ คือ 7
- 00001100 ผลลัพธ์ คือ 12

การดำเนินการลบเลขฐาน

- ให้ใช้วิธีการบวก
- ให้ตัวเลขหลังเครื่องหมายลบ คือ เลขจำนวนเต็มลบ
- ให้หา 2's complement ของเลขจำนวนเต็มลบนั่น เพราะคอมพิวเตอร์ จัดเก็บโดยใช้ระบบ 2's complement
- บวกตัวเลขที่เป็นตัวตั้งกับ 2's complement ของตัวบวก
- ตัวอย่าง 5 - 6 คือ การหาผลบวกของ $5 + (-6)$
- 0 0 0 0 0 1 0 1 คือ 5
- + 1 1 1 1 1 0 1 0 คือ 2's complement ของ -6
- 1 1 1 1 1 1 1 1 ผลที่ได้เป็น 2's complement ของ -1

การตรวจความผิดพลาด

- Overflow
 - ปรากฏการณ์ การล้น
 - เกิดขึ้นในกรณีที่ผู้โปรแกรมคำนวณแล้วได้ตัวเลขที่มีขนาดใหญ่กว่าความจุของหน่วยความจำ ส่วนใหญ่เกิดจากการบวกแล้วได้จำนวนเต็มที่เกินขนาดความจุ เช่น นำ -85 (1010 1011) กับ -95 (1010 0001) บวกกัน ผลลัพธ์จะเป็น -180 แต่เนื่องจาก พิสัยของตัวเลข 8 **bit** อยู่ในช่วง -128 ถึง 127 ดังนั้นผลลัพธ์ที่ได้จึงล้นออกมา
- Underflow
 - ปรากฏการณ์ น้อยเกินกว่าจะเก็บ
 - เกิดขึ้นในกรณีที่ผู้โปรแกรมคำนวณแล้วได้ตัวเลขที่มีค่าน้อยมากจนไม่สามารถนำไปเก็บในหน่วยความจำได้ ตรงข้ามกับ overflow

Real(จำนวนจริง)

- เป็นเลขทศนิยม floating point
- โดยทางวิทยาศาสตร์จะเขียนให้อยู่ในรูป normalized เช่น
 - 1.52×10^{15}
 - สำหรับเลข -0.001 สามารถเขียนให้อยู่ในรูป normalized ได้เป็น -1.0×10^{-3}
- จัดเก็บข้อมูลในหน่วยความจำในรูปเลขฐานสองเช่นเดียวกับ Integer
- เลขฐานสองที่ normalized เขียนคล้ายกับเลขฐานสิบ ตัวอย่างเช่น

$$1.1011 \times 2^5 = 110110$$

ตัวอย่างโปรแกรมจำนวนจริง L11.c

ตัวอย่างโปรแกรมจำนวนจริง L11.c

```
1  #include <stdio.h>
2
3  void main() {
4      int num1,num2,num3,sum;
5      float ave;
6
7      printf("Enter First Integer:");
8      scanf("%d",&num1);
9      printf("Enter Second Integer:");
10     scanf("%d",&num2);
11     printf("Enter third Integer:");
12     scanf("%d",&num3);
13     sum = num1+num2+num3;
14     ave = (float) sum/3;
15     printf("\nAverage of the three numbers is %6.4f.\n", ave);
16 }
17
```

Real(จำนวนจริง)

- การแปลงเลขทศนิยมฐานสิบ ให้เป็น เลขทศนิยมฐานสอง
- ตัวอย่าง จำนวนจริง 6.625 มีค่าเลขฐานสองเท่ากับ 110.101

$$1x2^2 + 1x2^1 + 0x2^0 + 1x2^{-1} + 0x2^{-2} + 1x2^{-3}$$

และมีค่าเท่ากับ

$$4 + 2 + 0 + \frac{1}{2} + \frac{0}{4} + \frac{1}{8} = 6.625$$

- เขียนให้อยู่ในรูป normalized $(110.101)_2 = (1.10101)_2 \times 2^2$
- $110110 = 1.101100 \times 2^5$

การแปลงเลขฐานสิบเป็นเลขฐานสอง

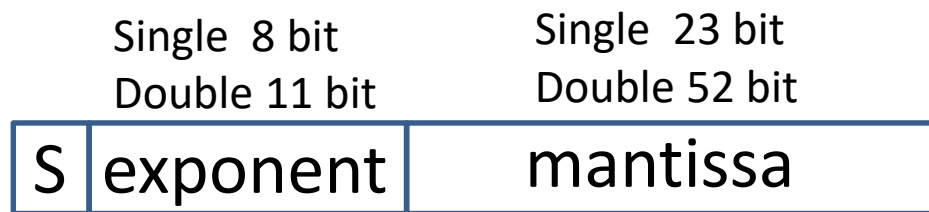
- ให้คูณเลขทศนิยมฐานสิบด้วย 2 จนกว่า เลขทศนิยมจะมีค่าเป็นศูนย์
ผลลัพธ์คือเลขจำนวนเต็ม(หน้าจุดทศนิยม) แต่ทุกครั้งจะเป็น
ค่าเลขฐานสองที่ได้
ตัวอย่างเช่น 0.125

$$\begin{array}{l} 0.125 \times 2 = 0.25 \\ 0.25 \times 2 = 0.50 \\ 0.5 \times 2 = 1.00 \end{array}$$

จำนวนจริง 0.125 มีค่าเลขฐานสองเท่ากับ 0.001

หน่วยความจำที่ใช้เก็บจำนวนจริง

- ตัวแปร float จะใช้รูปแบบ single precision ใช้หน่วยความจำขนาด 32 บิต
- ตัวแปร double จะใช้รูปแบบ double precision ใช้หน่วยความจำขนาด 64 บิต
- การจัดเก็บจะแบ่งออกเป็น 3 ส่วน
 - **บิตเครื่องหมาย(S)** มีค่าเป็น 0 สำหรับกรณีที่จำนวนจริงมีค่าบวก มีค่าเป็น 1 กรณีที่จำนวนจริงมีค่าเป็นลบ
 - **แมนทิสซ่า (mantissa)** ใช้เก็บตัวเลขที่อยู่หลังจุดทศนิยม
 - **เลขยกกำลัง (exponent)** ซึ่งจะมีทั้งยกกำลังเป็นบวก และเป็นลบ



Mantissa

- ก่อนจัดเก็บค่า mantissa ต้องถูก normalized ก่อน
- ตัวอย่างเช่น เลข $(13.625)_{10} = (1101.101)_2$
- ทำการ normalized จะได้ 1.101101×2^3
- โดยการจัดเก็บ เลข 1 ที่อยู่หน้าจุดทศนิยมจะถูกตัดทิ้ง เนื่องจากซ้ำซ้อน
- ให้นิสิตลองทำ normalized ของตัวเลขต่อไปนี้

$$0.001101_{\times 2^{-3}}, 1.001, 1000100.1 \times 2^6$$

- ลองเขียน เลขฐาน 2 ที่ normalized แล้วของ 15.35

1111, 01011
1-11101011 $\times 2^3$

อน

0.35
0.70
1.40 → 0.10
0.80 →
1.60
1.20

ตัดทิ้ง เนื่องจากซ้ำซ้อน

2) 15
2) 7 1
35) 3 1
1 1

~~0.35~~
0.70
1.40
0.80
1.60
1.20
0.40

Exponent

- การจัดเก็บในหน่วยความจำตามมาตรฐานของ IEEE 754 สำหรับตัวแปร float (single precision) ใช้เนื้อที่ 32 บิต
- ส่วนของ exponent ใช้เนื้อที่ 8 บิต
- ช่วงตัวเลขของ exponent ที่เป็นไปได้คือ -127 ถึง 128
- ค่าเลขยกกำลังที่จัดเก็บ = ค่ายกกำลังที่คำนวณได้จริง + ค่า bias(127)
- เพื่อปรับ exponent ให้เป็นตัวเลข unsigned

Exponent (E)	Adjusted (E + 127)	Binary Representation
5	+132	10000100
0	+127	01111111
-10	+117	01110101
128	+255	11111111
-1	+126	01111110

[http://cstl-csm.semo.edu/xzhang/Class%20Folder/CS280/Workbook HTML/FLOATING tut.htm](http://cstl-csm.semo.edu/xzhang/Class%20Folder/CS280/Workbook%20HTML/FLOATING%20tut.htm)

- ตัวอย่างการเก็บ 6.625 ในหน่วยความจำขนาด 32 บิต
- แปลงเป็นเลขฐานสองจะได้ 110.101
- เขียนให้อยู่ในรูป normalized $(110.101)_2 = (1.10101)_2 \times 2^2$
- จับเก็บในหน่วยความจำดังนี้

S	exponent	mantissa
0	1000 0001	101 0100 0000 0000 0000 0000

- ให้นิสิตจัดเก็บค่าต่อไปนี้

-1.01, 0.001101, 111001.01

How about?

- 0.0

Real(จำนวนจริง) ในภาษาซี

ประกอบด้วย

- float, double, long double
- ตัวแปรชนิด float มีจำนวนตำแหน่งทศนิยมน้อยกว่า double และ long double ตามลำดับ
- รูปแบบการพิมพ์ %f ใช้แสดงข้อมูลจำนวนจริง เช่น 300.545
- รูปแบบการพิมพ์ %e ใช้แสดงข้อมูลจำนวนจริงในรูปสัญกรณ์เชิงวิทยาศาสตร์ เช่น 3.00545e+02 และ 3.00545E+02 ตามลำดับ

ตัวอย่างโปรแกรมจำนวนจริง L12.c, L6.c

L12.c

```
1 #include <stdio.h>
2
3 void main() {
4     float s;
5
6     s = 300.545;
7     printf("\nScore are %f %6.4f %e %E\n", s, s, s,s);
8 }
9
```

L6.c

```
1  #include <stdio.h>
2
3  void main() {
4      float  tempFri;
5      double tempSat;
6      long double tempSun;
7
8      tempFri = 12.345;
9      printf("Friday temperature: %7.2f, ", tempFri);
10     printf("%10.3e,  %10.3E.\n",tempFri, tempFri);
11     tempSat = 12.465e-5;
12     printf("Saturday temperature: %7.2lf, ", tempSat);
13     printf("%10.5le,  %10.5lE.\n",tempSat, tempSat);
14     tempSun = 584.365E+17;
15     printf("Sunday temperature: %7.2Lf, ", tempSun);
16     printf("%10.3Le,  %10.3LE.\n",tempSun, tempSun);
17
18 }
19
```


ข้อมูลตัวอักษร



ตัวอักษรแต่ละตัวแทนด้วยรหัสไบนารี เช่น

อักขระ A แทนด้วย รหัสไบนารี 01000001

อักขระ T แทนด้วย รหัสไบนารี 01010100

‘A’ and ‘T’

เมื่อนำอักษรนี้ไปเก็บไว้ในหน่วยความจำขนาด 16 บิต จะได้

0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	0
															
A								T							

- ตัวอย่างโปรแกรม (L4.c)

L4.c

```
1 //from https://www.programiz.com/c-programming/c-input-output
2 #include <stdio.h>
3 int main()
4 {
5     char chr;
6     printf("Enter a character: ");
7     scanf("%c",&chr);
8
9     // When %c text format is used,
10    // character is displayed in case of character types
11    printf("You entered %c.\n",chr);
12
13    // When %d text format is used,
14    // integer is displayed in case of character types
15    printf("ASCII value of %c is %d.", chr, chr);
16    return 0;
17 }
18
```

ข้อมูล Boolean

- ข้อมูลประเภทบูลีน คือ ข้อมูลที่มีค่าเป็นจริง (true) หรือเท็จ(false)
- ในภาษา C การเรียกใช้ข้อมูลนี้ ต้อง include library `stdbool.h` ดังตัวอย่าง `L14.c`
- ข้อมูลประเภทบูลีนมักจะใช้กับคำสั่งที่มีการเช็คเงื่อนไขการทำงานของโปรแกรม เช่น คำสั่ง `if statement`
- ตัวแปรประเภทบูลีน อาจจะใช้ร่วมกับ `logical operator` ดังในตัวอย่าง `L15.c`
- การใช้ `relational operator` ทำให้เกิดข้อมูล Boolean
- หมายเหตุ `true` กับ `false` เป็น reserved word ในภาษา C

L14.c

```
1  #include <stdio.h>
2  #include <stdbool.h>
3
4  int main (){
5      bool one = true;
6      bool zero = 0;
7
8      if (one){
9          printf("Proposition is true.\n");
10     } else {
11         printf("Proposition is false.\n");
12     }
13     if (zero){
14         printf("Proposition is true.\n");
15     } else {
16         printf("Proposition is false.\n");
17     }
18
19     return 0;
20 }
```

Boolean logical operator in C

Boolean operator	meaning	example
&&	and	(a>0)&&(b<1)
	or	(x==y) (z==y)
!	not	!(x==0)

- ตัวอย่างการใช้งาน
if(x > max) max = x;
if(1 < i && i < 10)
- ตัวอย่างโปรแกรมบูตลิน (L5.C,L13.c, L15.c)

Relational operator

Relational operator	meaning	example
<	Less than	1<2
<=	Less than or equal	x<=y
>	Greater than	3>4
>=	Greater than or equal	x>=1
==	Equal	5==5
!=	Not equal	6!=6

L5.c

```
1 //from https://ubuntuforums.org/showthread.php?t=1826759
2 #include <stdio.h>
3 #include <stdbool.h>
4
5 bool function(){
6     static int i = 0;
7     return i++ % 2 == 0;
8 }
9
10 int main(){
11     int i;
12
13     for (i = 0; i < 10; ++i){
14         printf("function() returned: %s\n",
15             function() ? "true" : "false");
16     }
17     return 0;
18 }
19
```

L13.c

```
1  #include <stdio.h>
2
3  int main() {
4      int x,y;
5
6      printf("\nEnter an value of x:",);
7      scanf("%d",&x);
8      printf("\nEnter an value of y:");
9      scanf("%d",&y);
10     printf("Examples of logical expression\n");
11     printf("=====\\n");
12     printf(" x && y      : %d\\n",x&&y);
13     printf(" x || y      : %d\\n",x||y);
14     printf(" !x          : %d\\n",!x);
15     printf(" !y          : %d\\n",!y);
16     printf(" (x>0) && (y>0) : %d\\n", (x>0) && (y>0));
17     printf(" (x>0) || (y>0) : %d\\n", (x>0) || (y>0));
18
19 }
20
```


L15.c

```
1  #include <stdio.h>
2  #include <stdbool.h>
3  int main() {
4      bool a = 1;
5      bool b = 0;
6
7      // using "and" operator
8      if (a && b)
9          printf("True && False is True. (wrong)\n");
10     else
11         printf("True && False is False. (correct)\n");
12
13     // using "or" operator
14     if (a || b)
15         printf("True || False is True. (correct)\n");
16     else
17         printf("True || False is False. (wrong)\n");
18
19     return 0;
20 }
```