

lecture 1

- two's complement
- floating point numbers
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- hexadecimal <https://eduassistpro.github.io/>
[Add WeChat edu_assist_pro](#)

Car odometer (fixed number of digits)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\begin{array}{r} 0\ 0\ 6\ 0\ 9\ 9\ 9 \\ + \ 0\ 0\ 0\ 0\ 0\ 0\ 1 \\ \hline 0\ 0\ 0\ 1\ 0\ 0\ 0 \end{array}$$

9 9 9 9 9 9

+ Assignment Project Exam Help

— https://eduassistpro.github.io/

Add WeChat edu_assist_pro



99 99 99 ≡ - |

3 2 8 7 6 9



Assignment Project Exam Help

?



<https://eduassistpro.github.io/>



Add WeChat edu_assist_pro

$$\begin{array}{r} 328769 \\ + 671231 \\ \hline 000000 \end{array}$$

Assignment Project Exam Help

<https://eduassistpro.github.io/>

~~Add WeChat edu328769~~

If you know what "modular arithmetic" is (MATH 240), then you recognize this: addition of integers mod 10^6 .

Q: How to represent negative numbers in binary ?

A: Given an 8 bit binary number m ,
define $-m$ so that $m + (-m) = 0$.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\begin{array}{r} 00011010 \quad m \\ + \quad ? \\ \hline 00000000 \quad -m \end{array}$$

Two's complement representation of integers

Example: How to represent -26 ?

Use a trick!

Assignment Project Exam Help

<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro
+ ← invert bits

$$\begin{array}{r} 0000\ 11010 \\ + 11100\ 10 \\ \hline 111111111 \end{array}$$

$$\begin{array}{r}
 00011010 \quad n=2b \\
 + \underline{11100101} \quad \leftarrow \text{invert bits} \\
 \hline
 11111111
 \end{array}$$

+
 | ← add |

Assignment Project Exam Help
 00000000

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\begin{array}{r}
 11100101 \quad \leftarrow \text{inverted bits} \\
 + \underline{11100110} \quad \leftarrow \text{add } |
 \end{array}$$

← -2b

Another example: What is -0 ?

0 0 0 0 0 0 0 0 m = 0

| | | | | | | | invert bits

+ Assignment Project Exam Help^{add 1}

<https://eduassistpro.github.io/>

0 0 0 0 0 0 0 0
Add WeChat edu_assist_pro

+ 1 0 1 1 1 1 1 1
—————
0 0 0 0 0 0 0 0

We have verified that $-0 = 0$.

What about $m = 128$? What is -128 ?

$$\begin{array}{r} | 0 0 0 0 0 0 0 \\ + 0 | | | | | | \\ \hline \end{array}$$

$m = 128$

invert bits

add 1

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\begin{array}{r} 0 | | | | | | \\ + \\ \hline \end{array}$$

|

$$\begin{array}{r} | 0 0 0 0 0 0 \\ + \\ \hline \end{array}$$

$m = -128$

Thus, 128 is equivalent to -128.

binary

0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 1
0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1

"unsigned"

0
1
2
3
4

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

0 1 1 1 1 1 1
1 0 0 0 0 0 0 0

127

128

127
- 128 } ↗

1 1 1 1 1 1 0
1 1 1 1 1 1 1 1

254

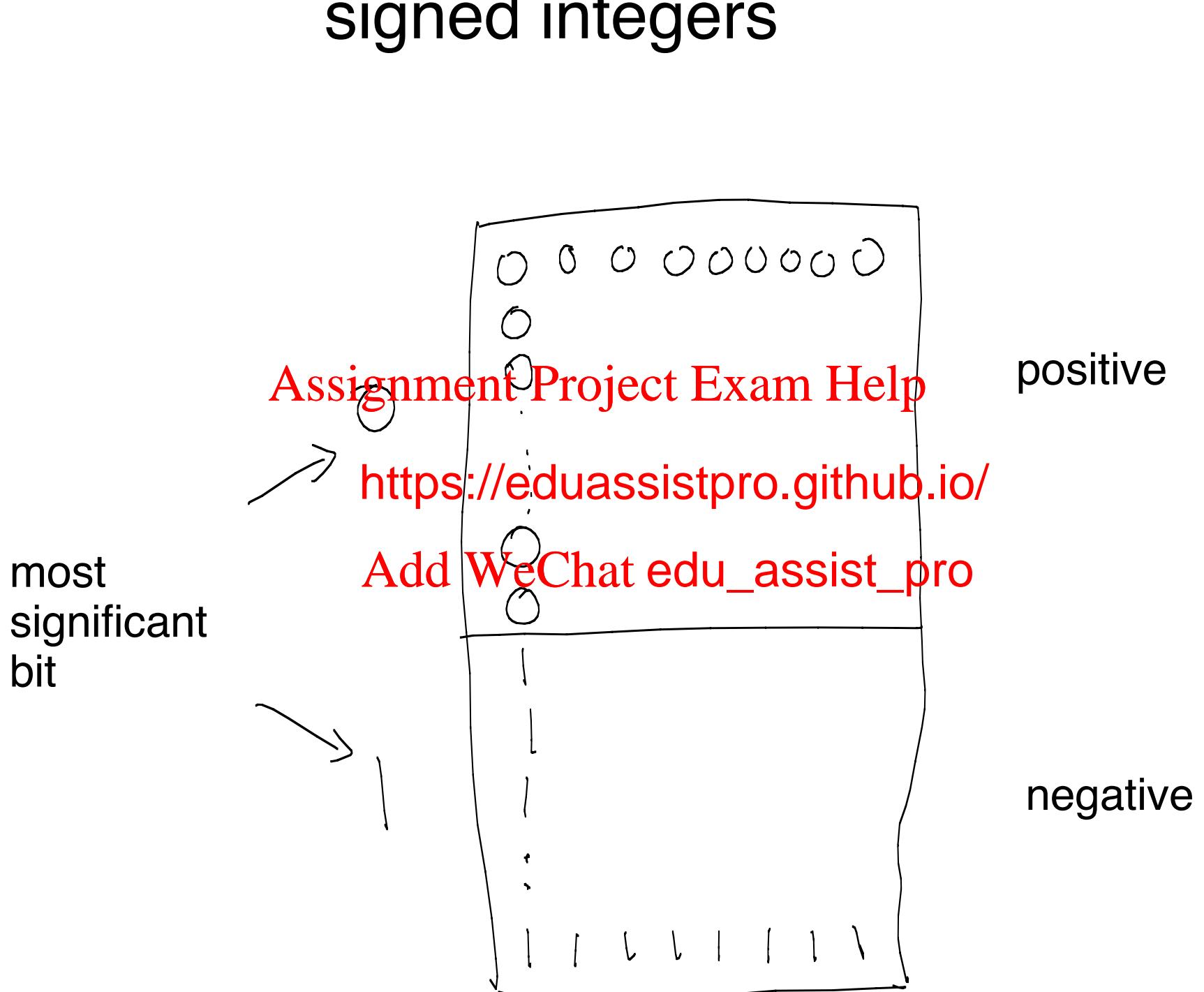
255

-2

-1

"signed"

signed integers



8 bit integers (unsigned vs. signed)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

n bits defines 2^n integers

unsigned

0

, | ,

Assignment Project Exam Help)

2^n

-

|

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

signed

-2^{n-1}

...

)

0

,

...

)

$n-1$

2^{n-1}

- |

Take n = 32.

The largest signed integer is $2^{31} - 1$.

$2^{10} = 1024 \sim 10^3$ = one thousand.

Assignment Project Exam Help

$2^{20} \sim 10^6$ million

Add WeChat edu_assist_pro

$2^{30} \sim 10^9$ = one billion

$2^{31} \sim 2,000,000,000$ = two billion

Java Example

```
int j = 4000000000; // 4 billion > 2^31
```

This gives a compiler error. "The literal of type int is out of range."

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro
System.out.println(2 * j)

// This prints out -294967296.

// To understand why these particular digits are printed, you
// would need to convert 4000000000 to binary, which I don't
// recommend.)

lecture 1

- two's complement
 - floating point numbers
 - hexadecimal
- Assignment Project Exam Help
<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro

Floating Point

"decimal point"



$$26.375 = 2 \times 10^1 + 6 \times 10^0 + 3 \times 10^{-1} + 7 \times 10^{-2} + 5 \times 10^{-3}$$

Assignment Project Exam Help

<https://eduassistpro.github.io/>

"binary point" Add WeChat edu_assist_pro

$$\begin{aligned} (11010.011)_2 &= 2^4 + 2^3 + 2^1 + 2^{-2} + 2^{-3} \\ &= 16 + 8 + 2 + 0.25 + 0.125 \\ &= 26.375 \end{aligned}$$

Convert from binary to decimal

We must use both positive and negative powers of 2.

$$\begin{array}{r} \overbrace{}^n \\ -1 \\ \hline \end{array} \quad \begin{array}{r} \overbrace{2}^n \\ .5 \\ \hline \end{array}$$

Assignment Project Exam Help

- 3 .125
- 4 https://eduassistpro.github.io/

- 5 Add WeChat edu_assist_pro

- 6 .015625
- 7 , 0078125
: etc.

Sum up the contributing 1 bits as on previous slide.

How to convert from decimal to binary ?

$$26.375 = (\underline{\quad ? \quad} \cdot \underline{\quad ? \quad})_2$$

To find the bits for the positive powers of 2, use the algorithm from ~~Assignment Project Exam Help~~ ("Repeated Division").

<https://eduassistpro.github.io/>

~~Add WeChat edu_assist_pro~~

$$\begin{array}{r} \overline{m} \\ 26 \\ \hline 13 & 0 \\ 6 & 1 \\ 3 & 0 \\ | & 1 \\ 0 & 1 \end{array} \Rightarrow \begin{array}{l} 2^6 \\ = (11010)_2 \end{array}$$

What about negative powers of 2 ?

In general, note that multiplying by 2 shifts bits to the left
(or shifts binary point to the right)

Assignment Project Exam Help

Example:

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\begin{aligned} & \left(11010.011\right)_2 \times 2 \\ = & \left(11010\ 0.11\right)_2 \end{aligned}$$

Similarly....dividing by 2 and not ignoring remainder shifts bits to the right (or shifts binary point to the left)

Assignment Project Exam Help

$$\left(\begin{smallmatrix} 1 & 1 & 0 & 1 & 0 & . & 0 & 1 & 1 \end{smallmatrix} \right)_2 / 2$$

https://eduassistpro.github.io/
Add WeChat edu_assist_pro

$$= \left(\begin{smallmatrix} 1 & 1 & 0 & 1 & 0 & . & 0 & 0 & 1 & 1 \end{smallmatrix} \right)_2$$

For the negative powers of 2, use "repeated multiplication"

.375

$$= .375 \times 2^1 \times 2^{-1}$$

Assignment Project Exam Help

$$= \frac{75}{100} \times 2^{-2}$$

<https://eduassistpro.github.io/>

$$= \frac{1.5}{2} \times 2^{-3}$$

Add WeChat edu_assist_pro

$$= 3.0 \times 2^{-4}$$

convert
decimal
to binary

$$\hookrightarrow = (11)_{2^3} \times 2^{-3}$$

$$= (.011)_2$$

A more subtle example:

$$19.243 = (?)_2$$

First, find the bits for the positive powers of 2 using "repeated division" (last lecture).

Assignment Project Exam Help

m

19

4 <https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

9

1

$$\therefore 19 = (1001)_2$$

4

1

2

0

1

0

0

1

Then find the bits for the negative powers of 2 using repeated multiplication.

$$\begin{array}{r} \cdot 243 \\ = 243 \times 2^1 \times 2^{-1} \\ \text{Assignment Project Exam Help} \end{array}$$

<https://eduassistpro.github.io/>
≡ (Add WeChat ~~edu_assist_pro~~)

≡ ?

Then find the bits for the negative powers of 2 using repeated multiplication.

$$\begin{aligned} & \cdot 243 \\ &= (0)_2 \cdot 486 \times 2^{-1} \\ &= (00)_2 \cdot 972 \times 2^{-2} \\ &= (001)_2 \cdot 944 \times 2^{-3} \\ &= (0011)_2 \cdot 888 \times 2^{-4} \\ &= (0011)_2 \end{aligned}$$

Assignment Project Exam Help
<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro

Thus $(.243)_{10} = (.0011)_2 + \sum_{i=-5}^{-\infty} b_i 2^i$

Note the summation is over bits b_i from -5, -6, ..., -infinity.

$$19.243 = (10011.0011 \underline{\quad})_2$$

We cannot get an exact representation using a finite number of bits for this example.

Assignment Project Exam Help

Can we say anyt <https://eduassistpro.github.io/> out what happens ?

Add WeChat edu_assist_pro

This will repeat over and over again.

$$\cdot (05)_{10} = (0)_2 \cdot 1 \times 2^{-1}$$

$$= (00)_2 \cdot 2 \times 2^{-2} - 3$$

$$= (000)_2 \cdot \text{Assignment Project Exam Help}$$

<https://eduassistpro.github.io/>

$$= (0000)_2 \cdot 8 \times 2^{-5} \text{ Add WeChat edu_assist_pro}$$

$$= (00000)_2 \cdot 6 \times 2^{-6}$$

$$= (000001)_2 \cdot 2 \times 2^{-6}$$

$$= 00 \underline{0011} 0011 \underline{0011} \quad \text{etc.}$$

When we convert a floating point decimal number with a finite number of digits into binary, we get:

- a finite number of non-zero bits to left of binary point
- an infinitely repeating sequence of bits to the right of the binary point

Assignment Project Exam Help

Why ?

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

[Note: sometimes the infinite number of repeating bits are all 0's, as in the case of 0.375 a few slides back.]

Recall previous example...

$$\begin{aligned} & \cdot \quad 243 \\ = & (0)_2 \cdot 486 \times 2^{-1} \\ = & (00)_2 \cdot 972 \times 2^{-2} \\ = & (001)_2 \cdot 944 \times 2^{-3} \\ = & (0011)_2 \cdot 888 \times 2^{-4} \\ = & (00110)_2 \cdot 833 \times 2^{-5} \\ = & \dots \text{etc} \end{aligned}$$

Eventually, the three digits to the right of the decimal point will enter a cycle that repeats forever. This will produce a bit string that repeats forever.

Hexadecimal

Hexadecimal (base 16)

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
a	1010
b	1011
c	1100
d	1101
e	1110
f	1111

Assignment Project Exam Help

Writing down long strings of bits is awk and error prone.

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

Hexadecimal simplifies the representation.

Examples of hexadecimal

1) 0010 1111 1010 0011

2 f a 3

Assignment Project Exam Help

We write 0x <https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

2) 101100

We write 0x2c (10 1100), not 0xb0 (1011 00)