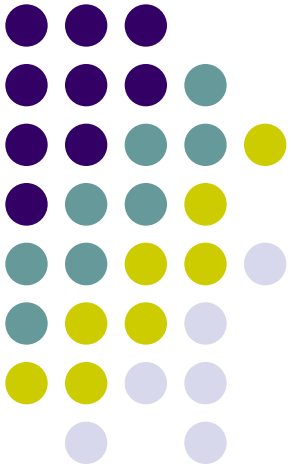
CS 460

Programming Languages

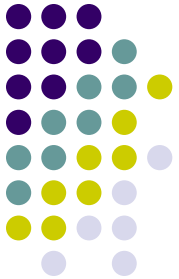
Fall 2021

Dr. Watts

(11 October 2023)

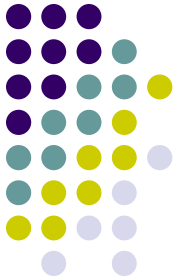


Project 1



- Questions? (By email or on paper)

Half Precision (float16)

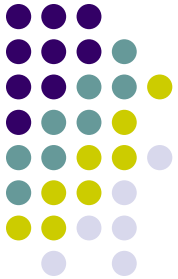


- Half precision floating point representation requires a 16 bit word, which may be represented as numbered from 0 to 15, left to right. The first bit is the sign bit, S, the next five bits are the exponent bits, 'E', and the final 10 bits are the fraction 'F':

```
S EEEEE FFFFFFFFFF
0 1   5 6        15
```

- The value V represented by the word may be determined as follows
- Sign
  - Sign = 0 is positive
  - Sign = 1 is negative
- Exponent
  - Biased (15)
  - 00000 – 11111
- Fraction

Half Precision Example 4

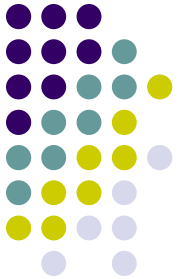


- Example: 5.20
- Whole: 5 : 101
- Fraction: .2 : 0011001100110011001100
- Unnormalized: 101.0011001100110011001100
- Exponent: 2
- Normalized: 1.0100110011001100110011
- Next is 0

5.2

Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	0	1	0	0	0	1	0	1	0	0	1	1	0	0	1	1

Half Precision Example 5

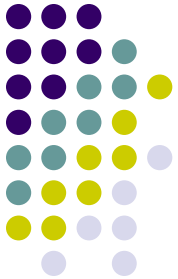


- Example: 12.20
- Whole: 12 : 1100
- Fraction: .2 : 0011001100110011
- Unnormalized: 1100.0011001100110011
- Exponent: 3
- Normalized: 1.1000011001100110011
- Next is 1

12.2

Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	0	1	0	0	1	0	1	0	0	0	0	1	1	0	1	0

Half Precision Example 6

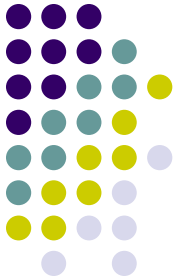


- Example: 25.20
- Whole: 25 : 11001
- Fraction: .2 : 0011001100110011001100
- Unnormalized: 11001.0011001100110011001100
- Exponent: 4
- Normalized: 1.10010011001100110011001100
- Next is 1

25.2

Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	0	1	0	0	1	1	1	0	0	1	0	0	1	1	0	1

Half Precision Example 7

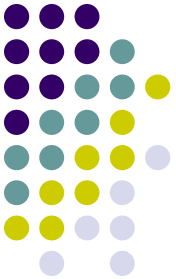


- Example: 37.20
- Whole: 37 : 100101
- Fraction: .2 : 0011001100110011001100
- Unnormalized: 100101.0011001100110011001100
- Exponent: 5
- Normalized: 1.00101001100110011001100
- Next is 0

37.2

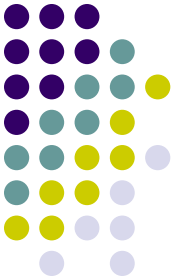
Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	0	1	0	1	0	0	0	0	1	0	1	0	0	1	1	0

Let's go the other way!



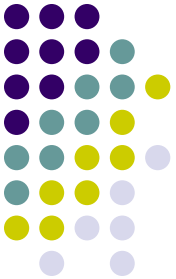
Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0

Let’s go the other way!

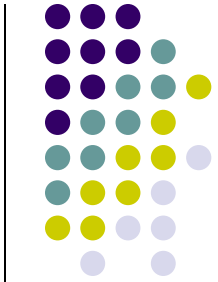


Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit value	0	1	0	1	1	1	0	0	1	1	0	0	1	1	0	1

Some Half-precision values



Binary	Hex	Value	Notes
0 00000 0000000000	0000	0	
0 00000 0000000001	0001	$2^{-14} \times (0 + \frac{1}{1024}) \approx 0.000000059604645$	smallest positive subnormal number
0 00000 1111111111	03ff	$2^{-14} \times (0 + \frac{1023}{1024}) \approx 0.000060975552$	largest subnormal number
0 00001 0000000000	0400	$2^{-14} \times (1 + \frac{0}{1024}) \approx 0.00006103515625$	smallest positive normal number
0 01101 0101010101	3555	$2^{-2} \times (1 + \frac{341}{1024}) \approx 0.33325195$	nearest value to 1/3
0 01110 1111111111	3bff	$2^{-1} \times (1 + \frac{1023}{1024}) \approx 0.99951172$	largest number less than one
0 01111 0000000000	3c00	$2^{0} \times (1 + \frac{0}{1024}) = 1$	one
0 01111 0000000001	3c01	$2^{0} \times (1 + \frac{1}{1024}) \approx 1.00097656$	smallest number larger than one
0 11110 1111111111	7bff	$2^{15} \times (1 + \frac{1023}{1024}) = 65504$	largest normal number
0 11111 0000000000	7c00	∞	infinity
1 00000 0000000000	8000	-0	
1 10000 0000000000	c000	-2	
1 11111 0000000000	fc00	$-\infty$	negative infinity



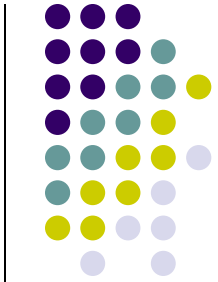
```
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;

int main ()
{
        float value = 0;
        do
        {
                cout << "Please enter a floating point value (0 to quit): ";
                cin >> value;
                cout << "  value: " << fixed << showpoint << setprecision (30)
                     << value << endl;
                float rounded = (round (value * 100)) / 100.0;
                cout << "rounded: " << fixed << showpoint << setprecision (30)
                     << rounded << endl;
        } while (value != 0);
        return 0;
}
```

Other Numeric Types

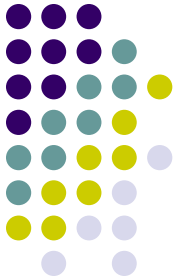
- Arbitrary precision
- Fraction type
- User defined types

String types



- C style strings
- C++ strings

Composite Data Types



- Array
- Struct
- Class
- Union

Contents