NSD midterm (Rm EC114) 8:00am-9:00am 7th November

Name: _____ Student ID: _____

No notes, calculators, or other aids are allowed. Read all instructions carefully and write your answers in the space provided. To receive full credit, you must show all of your work. The computer architecture used in the questions is assumed to be x86 and the runtime environment is the homework CI setup. If arriving the classroom on time, you may use up to 60 minutes. One arriving after the first one turning in the exam sheet will not be admitted.

There are 30 points in total.

1. (3 points) Write the output of the following C++ code:

```
1  #include <iostream>
2  #include <cstdint>
3  int main(int, char **)
4  {
5      long sint = -(1<<8);
6      unsigned long uint = (1<<8);
7      std::cout << "sint: " << sint << std::endl;   // Line 1
8      std::cout << "uint: " << uint << std::endl;   // Line 2
9      if (sint > uint)
10     { std::cout << "sint > uint" << std::endl; }  // Line 3
11     else if (sint == uint)
12     { std::cout << "sint == uint" << std::endl; } // Line 3
13     else
14     { std::cout << "sint < uint" << std::endl; }  // Line 3
15     return 0;
16 }
```

It's built with:

```
g++ -O3 q1.cpp -o q1
```

Line 1 :

Line 2 :

Line 3 :

2. (3 points) What is the value of the following expression (show as much detail as you can):

```
0x1 << sizeof(int)*5
```

Value:

3. IEEE 754 single-precision floating-point uses 32 bits (4 bytes). The first (lowest) 23 bits are fraction. The following 8 bits are exponent with the bias 127 (`0111 1111`). The last (highest) bit is sign; 0 is positive while 1 is negative.

   Consider a decimal number 2.75 and write it in the base of 2:

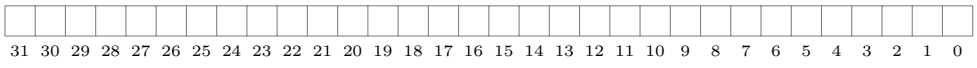
$$\begin{aligned} 2.75 &= 2 + \frac{1}{2} + \frac{1}{2^2} = 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} \\ &= (10.11)_2 = (1.011)_2 \times 2^1. \end{aligned}$$

   The bit fields (from high to low) for its IEEE 754 single-precision floating-point are:

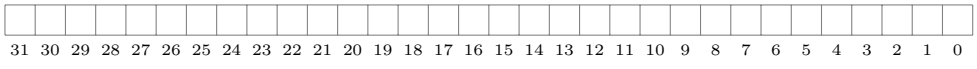
sign (1 bit)	exponent (8 bits)	fraction (23 bits)
`0`	`1000 0000`	`011 0000 0000 0000 0000 0000`

   (a) (6 points) Write the bit fields of IEEE 754 single-precision floating-point value for 4.2 and -4.2 (decimal). Show as much detail as you can.

4.2:

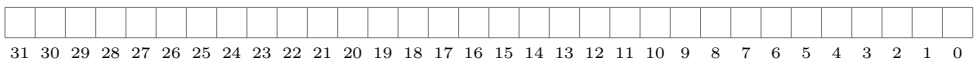


-4.2:



   (b) (3 points) Write the bit-wise XOR of the bit fields of the two single-precision floating point values (4.2 and -4.2).

Answer:



4. (3 points) Write the output of the following C++ code:

```
#include <iostream>
#include <cmath>
#include <limits>
int main(int, char **)
{
    float v1;
    v1 = 0.3;
    std::cout << "result: " << v1/0 << std::endl;
    v1 = M_PI/2;
    std::cout << "std::asin(M_PI/2): " << std::asin(v1) << std::endl;
    v1 = std::numeric_limits<float>::max();
    std::cout << "std::numeric_limits<float>::max() * 2: " << v1 * 2
              << std::endl;
    return 0;
}
```

   1. `result:`

   2. `std::asin(M_PI/2):`

   3. `std::numeric_limits<float>::max() * 2:`

5. (3 points) Write the numpy dtype names (strings) corresponding to the following C++ fundamental types:

   1. `char:`

   2. `unsigned int:`

   3. `double:`

6. (3 points) Consider a 3×4 matrix:

$$\mathrm{A}_{3\times4} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix}$$

   Write down how the elements are ordered in sequence in memory when the matrix is column-major.

   Answer:

7. (6 points) Assume a main memory of 64 (2^6) bytes and a direct-mapped cache of 8 (2^3) bytes. The cache is initialized as empty.

   Fill the 12 empty cells in the following table. The "hit or miss" column may use one of the 3 possible value: "cold miss", "hit", or "conflict miss".

access #	memory addr	hit or miss	cache block addr
1	10110	cold miss	110
2	11010		
3	10110		
4	11010		
5	10000		
6	10010		
7	11010		