Andrew Brodsky | SID: 107217786 | Assignment 1 | Prof: Maciej Zagrodzki

1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Type** | **Minimum** | **Maximum** | **Formula(s)** |
| int8 | -128 | 127 | Min: -27; Max: 27-1  Range: 28 |
| int16 | -32,768 | 32,767 | Min: -215; Max: 215-1  Range: 216 |
| int32 | -2,147,483,648 | 2,147,483,647 | Min: -231; Max: 231-1  Range: 232 |
| int64 | -9,223,372,036,854,775,808 | 9,223,372,036,854,775,807 | Min: -263; Max: 263-1  Range: 264 |
| uint8 | 0 | 255 | Max: 28-1  Range: 28 |
| uint16 | 0 | 65,535 | Max: 216-1  Range: 216 |
| uint32 | 0 | 4,294,967,295 | Max: 232-1  Range: 232 |
| uint64 | 0 | 18,446,744,073,709,551,615 | Max: 264-1  Range: 264 |

2.

1. Is ​fix(6.5)​ the same as ​floor(6.5)​? **YES**
2. Is ​fix(3.3)​ the same as ​fix(-3.3)​? **NO**
3. Is ​fix(4.2)​ the same as ​floor(4.2)​? **YES**
4. Is ​fix(-5.3)​ the same as ​floor(-5.3)​? **NO**
5. Is ​fix(-7.2)​ the same as ​ceil(-7.2)​? **YES**
6. Is ​round(-2.4)​ the same as ​floor(-2.4)​? **NO**
7. Is ​round(-8.4)​ the same as ​ceil(-8.4)​? **YES**

3.

>> 3\9 **=3**  
>> - 5 ^ 2 **=-25**  
>> (-5) ^ 2 **=25**  
>> 10-6/2+3 **=10**  
>> 3 == 5 + 2 **=0 (logical)**  
>> 'b' >= 'c' – 1 **=1 (logical)**  
>> 7 == 6 + 1 **=1 (logical)**  
>> (7 == 6) + 1 **=1**  
>> xor(5 < 6, 8 > 4) **=0 (logical)**  
>> xor('c' == 'd' - 1, 2 > 4) **=1 (logical)**

4.

1. real number in the range (0, 25) **>> 25\*rand()**
2. real number in the range (20, 50) **>>30\*rand()+20**
3. integer in the inclusive range from 1 to 10 **>>randi([1 10], 1)**
4. integer in the inclusive range from 0 to 10 **>>randi([0 10], 1)**
5. integer in the inclusive range from 50 to 100 **>>randi([50 100], 1)**

5.

1. Create two variables ​*x*​ and ​*y*​ that will store positive or negative integers.

**>> x=int8(0)**

**>>y=int8(0)**

1. Return ​true​ if the value of ​*x*​ is greater than five or if the value of ​*y*​ is less than ten, but not if both of those are ​true​.

**>>xor(x>5, y<10)**

6. Using only the integers 2 and 3, write as many expressions as you can that result in 9.

1. **>>3\*3**
2. **>>3\*3\*3/3**
3. **>>(3/2)\*2\*3**
4. **>>(2\*2\*2\*2+2)/2**
5. **>>floor(3\*2\*2-2\*3/2)**
6. **>>fix(2\*2\*2+3/2)**
7. **>>round((2/3)\*(3\*3+2\*2))**
8. **>>ceil(3/2\*3\*2-2/3)**
9. **>>fix(randi([27,29],1)/3)**
10. **>>ceil(rand()+2^3)**