

Считайте что это и есть тест

задачи на вещественные числа (не затрагивая прогу)

1. перечислите все необычные состояния вещественные числа float типа, когда они достигаются. на примере real32, real64

(a) NaN:sNaN и qNaN (quiet NaN, signing NaN)

(b) +inf, -inf,

(c) +0, -0

пример в half precision

+inf	0 11111 00 0000 0000	− inf	1 11111 00 0000 0000
sNaN	0 11111 00 0000 0001	qNaN	0 11111 10 0000 0000
+0	0 00000 00 0000 0000	-0	1 00000 00 0000 0000

2. заполните табличку числом разрядов

стандарт	размер	бит знака	порядок	мантисса	bias
half precision	16	1	5	10	15
single precision	32	1	8	23	127
double precision	64	1	11	52	2047
quad precision	128	1	15	112	32767
extended precision	80	1	15	64	32767

3. как громкий NaN (sNaN) переделать в тихий NaN (qNaN): Старший бит мантисы поставить на 1
4. что выдаст sqrt(-1.0f) ;;C код - лучший ответ sNaN - так как qNaN ставит только программист по своему усмотрению ответ NaN
5. В каких случаях float может считаться денормализованным - если число $1.0 * 2^{bias}$ пример в половинной точности 2^{-15} , хоть само число ещё вроде можно представить но так как все нули это 0, то мы будем записывать уже денормализованное число, те E=0 а старший бит Мантиссы есть 1 а не 1/2
ещё раз 2^{-15} будет записано как 0 00000 10 0000 00000 числа меньше также

6. переведите в число единичной точности

+0: 00000000

86.125

(a) $86.125_{10} = 1010110.001_2 = 1.010110001_2 * 2^6$,

(b) $bias = 2^{8-1} - 1 = 127$, $E = 6 + 127 = 133 = 10000101_2$

(c) итого запись: $0|10000101|01011000100000000000000_2 = 42AC4000_{16}$

196.75

приведу способ быстро проверить себя убедившись что числа действительно так хранятся в ассемблере, заодно закинув макросы:

```
include console.inc
include helpful.inc
```

```
.data
    pr real4 196.75
.code
Start:
    outbin pr
    newline
    outhex pr
    exit
end Start
```

вспоминаем как нас научил Алексеев:

$$\frac{1}{3}$$

1	0,
1*2=2, 2 mod 3 = 2	2 div 3 = 0
2*2=4, 4 mod 3 = 1	4 div 3= 1

$$E = -2 + 127 = 125 = 01111101_2$$

получаем: 0|01111101|010101010101010101010|**101** - необходимо округлить к ближайшему чётному
мантиссу $0/1 \rightarrow 0$

ответ: $0\ 0111\ 1101\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101_2 = 3EAAAAA_{16}$

$$-0 = 800000000_{16}$$

$$20 * 2^{-128} = 1.01_2 * 2^4 * 2^{-128}$$

порядок = -124: -124+127=3

ответ: 0 0000 0011 0100 0000 0000 0000 0000 000

7. сложите два числа half precision:

$$890.5 + 10.5625$$

окомментируйте все сдвиги поэтапно

переведём числа в биты:

$$890.5 = 1.1011110101_2 * 2^9 = 0\ 11000\ 10\ 1111\ 0101_2 \text{ (a)}$$

$$10.5625 = 1.0101001_2 * 2^3 = 0\ 10010\ 01\ 0100\ 1000_2 \text{ (b)}$$

разница между экспонентами 11000_2 и $10010_2 = 6$

сдвигаем мантиссу числа (b) на 6 стандарт IEEE-754 требует чтобы операции происходили с большей точностью

1.1011110101000 (a)

0.0000010101001 (b)

1.1100001010001 (a+b)

1.1100001010001 - нормализуем (уже нормализовано экспонента остаётся как у (а))

округляем на лекциях мы рассматривали стандартное банковское к ближайшему чётному

1100001010 0 - первые 10 чисел после точки и 11 число 00 $\rightarrow 0$

получится число: $0\ 11000\ 1100001010 = 901$

8. решите уравнение считая что числа округление банковское single precision $98.3125 + X = 98.3125$

$$98.3125 = 1100010.0101 = 1.1000100101 * 2^6$$

его запись будет $0_10000101_100010010100000000000000_2$ чтобы число не поменялось нужно прибавить что то маленькое по модулю чтобы после округления оно осталось таким же

[illegible]

так как округление банковское а последний разряд 0 то при любом исходе округляться будет в меньшую сторону $-23+6=-17 \rightarrow$

$$-2^{-17} < X < 2^{-17}$$

НЕРАВЕНСТВА СТРОГИЕ!!

решите это же самое уравнение уже в half precision, сравните результат

$$98.3125 = 1100010.0101 = 1.1000100101 * 2^6$$

его запись будет 0 10101 1000100101

аналогично несложно понять что искомый X по модулю меньше $0.0000000001_2 * 2^6$ (10 знаков после точки), но последний бит $\neq 0 \rightarrow$

возможно округление для это нужно чтобы число было больше или равен $0.000000000001_2 * 2^6$ (11 знаков после точки) делаем вывод $-11+6=-5$

$$-2^{-5} < X < 2^{-5}$$

9. достижение наибольшего и наименьшего значения float
все единицы, но дес Порядок, иначе NaN (qNaN) - min
все единицы, кроме 1 бита = 0 и дес Порядок, иначе NaN (qNaN) - max

Ниже ответы на следующую часть

задачи на ассемблере

1. Укажите разрядность XMM регистра, сколько их
128 бит 8 штук
2. Согласно соглашению (cdecl, stdcall) как передаётся вещественное число
в вершине стека FPU в st0
3. реализовать сложение, вычитание чисел (дополнительно умножение, деление) одинарной и двойной сложности используя SSE2
4. Сложить NaN с любым другим числом
5. Сравнить поведение qNaN и sNaN Если переопределить обработку прерывания то будет видно а так нам незаметно;
6. Приведите операции что в результате дадут (real4 (single precision)):
 - (a) +inf
 - (b) -inf
 - (c) NaN
 - (d) +0
 - (e) -0

смотреть код