20210815\_JaeHyeon\_Jeong

1. int negate(int x)

기본적으로 negate를 하기 위해 2의 보수를 이용하기 때문에 1의보수(~) +1을 사용하여 return으로 ~x+1을 하여 negate를 구현하였다.

2. int isLess(int x, int y)

isLess 즉 x<y가 되기 위해서는 조건이 두가지가 존재하는데 첫번째는 간단하게 x가 음수 y가 양수면되는데 음수일 때 msb가 1 양수일 때 msb가 0이기 때문에, x&~y를 하였고 이어서 두번째 조건은 두 수의 부호는 같으나 연산 결과가 음수인 경우인데 부호가 같은 경우 x^y시 0이 되므로 ~(x^y)를 하고 이어서 x<y이므로 x-y<0 즉 x~y+1이 음수이므로 ~(x^y)&(x~y+1)을 해주는데 이 두가지 |로 연결하고 >>31을 하여 msb를 추출해주면 그값이 1일 때 isless가 성립할 수 있도록 구현하였다.

3. unsigned float\_abs(unsigned uf)

floating num을 절댓값을 취해주기 위해 먼저 exp와 frac을 추출한다. Uf>>23&0xFF로 exp를 추출하고, uf&0x7FFFFF로 frac을 추출한다 이때 NaN의 경우 uf를 다시 리턴을 해야하므로 if문을 이용하여 exp=0xFF이면서 frac가 0x0이 아닐경우 uf을 하도록 하고, 일반적인 경우에는 (exp<<23)|frac을 리턴하여 msb가 0인 양수로 만들어준다.

4. unsigned float\_twice(unsigned uf)

먼저 uf를 받으면 2배를 하기 전에 s,exp,frac을 분리해야한다. (uf>>31)&1로 사인을 추출해주고 ((uf>>23)&0xFF)로 exp를 추출하고 (uf&0x7FFFFF)으로 frac을 추출한다. 이때, 문제에 조건에 따라 exp가 0xFF인 경우 uf를 리턴하도록 if문을 사용하였고, exp가 0인 denormalized value에 대해서는 exp를 증가시킬 수 없으므로 frac을 1칸 쉬프트 레프트를 해준다. 특히 exp가 0xFF-1일 때에는 exp를 1 증가시키면 0xFF가 되기 때문에 exp=0xFF로 하고, frac을 0으로 만들어준다. 마지막으로 이외에 상황에는 exp를 1 증가시켜 twice를 만들어준다. 이렇게 만든 결과로 (s<<31)|(exp<<23)|frac)를 하여 결과를 모아서 twice된 uf를 리턴해주도록 구현하였다.

5. unsigned float\_i2f(int x)

Integer를 floating point number로 표현해주기 위해서 먼저 int x를 받아오는데, 기본적으로 32비트이기 때문에 exp는 8비트 즉 bias는 127이다. 먼저 x의 부호를 따지기 위해 x와 (0x80000000)를 &로 연산하여 부호를 나타내는 s에 저장한다. 그리고 exp =.bias(127)+31이므로 158을가짖고 있도록 하고 frac은 일단 기본값으로 0을 주었다. 먼저 x가 0인경우 if문을 이용하여 0을 리턴하도록 하였고, x가 tmin(0x7FFFFFFF+1)인 경우 frac=0x0 s=0x1, exp=158 을 모아서 s<<31 | exp <<23 | frac 을 리턴하도록 하였다. 이어서 x가 음수일 경우 x=-x로 양수로 만들어주어 frac과 exp를 계산하도록 하였다 이어서, while문을 이용하여 x를 1씩 쉬프트 시키면서 msb가 처음으로 1이 될 때 마무리되도록 하였는데 이때 exp를 1씩 감소시키고, x를 레프트 쉬프트 1을 해주면서 진행하였다. 그 결과 msb부분에 1이 오기전 단계에서 while문이 끝나기 때문에 추가적으로 exp-=1과 x<<=1을 해주었다. 그결과 나온 x값은 msb가 1이고 그 뒤로 frac 파트가 존재하는데 이부분을 0x7FFFFFFF과 &연산을 하고 exp파트의 크기인 8비트 만큼 라이트 쉬프를 하여 frac을 구성하였다. 마지막으로 frac으로 x를 표현할 수 없을 때 생략되는 곳을 라운딩 해줘야한다. 이때 x가 127과 &한 값이0보다 크거나 frac의 첫번째 비트가 1이면서 x&12을 만족하면 frac을 1증가시키면서 라운딩을 한다. 이후 사인을 나타내는 S와 exp을 23만큼 레프트 쉬프트 한 값과 frac을 더해준 값을 더해주고 리턴해주며 해당 함수를 구현하였다.

6. int float\_f2i(unsigned uf)

Floating point number를 고치는 것이므로 먼저 uf를 사인,exp,frac으로 나눠서 받도록 한다. (uf >> 31)&0x1를 하여 부호 값을 받도록 하고 (uf >> 23)&(0xFF)으로 8비트의 exp값을 추출하고, (uf & 0x7FFFFF)를 하여 23비트의 frac을 추출한다. 이어서 exp가 8비트 이므로 bias의 값은 2^(8-1)-1인 127을 가지도록 하고 normalized value 계산을 위해 e=exp-bias로 하도록 하고, M도 (frac + 0x800000u)을 하여 normalized 계산을 하도록한다. 이렇게하는 이유는 if문을 이용하여 denormalized value를 따로 리턴하기 때문에 나머지는 normalized value라서 정하였다. 먼저 exp의 값이 bias보다 작으면 0을 리턴하도록 만들었고, 이는 E의 값이 0보다 작아서 0으로 수렴함을 의미한다. 이어서 if문으로 e의 값이 31보다 크게 되면 문제의 조건에따라 0x80000000u를 리턴하도록 구현하였다. 이어서 e의 값이 23보다 크거나 같다면 normalized value인 m을 e-23만큼 쉬프트 레프트를 해주어 result에 저장하고, 반대라면 m을 쉬프트 라이트로 23-e만큼 해주어 result에 저장한다. 마지막으로 사인을 나타내는 s가 1일 경우 -result를 리턴해고 아닐경우 그대로 리턴하여 floating point num을 알맞게 int로 리턴하도록 구현하였다.