In [1]:

```
import numpy as np
dict_data={"alpha":0.0072973525693,"c":29979245800,"G":6.6743*10**(-8),"h
for i in dict_data.keys():
    dec_num=dict_data[i]
    dec_l=int(np.floor(abs(dec_num)))
    dec_r=abs(dec_num)-dec_l
    def dec_l2b(num,l=[]):
        l.append(num%2)
        num=num//2
        if num!=0:
            dec_l2b(num,l)
        if num==0:
            l.reverse()
        return l
    def dec_r2b(num,l=[],count=0):
        d=num*2
        if d>=1:
            d1=1
            num=d-1
        else:
            d1=0
            num=d
        l.append(d1)
        count=count+1
        if count!=100:
            dec_r2b(num,l,count)
        return l
    a=dec_l2b(dec_l,[])
    b=dec_r2b(dec_r,[])
    c=a+b
    for j in range(len(c)):
        if c[j]==1:
            mantissa=c[j:]
            break;
    mantissa=mantissa[1:24]
    if dec_l!=0:
        exp=len(a)-1+127
    else:
        exp=-j+127
    exp_bin=dec_l2b(exp,[])
    e_size=len(exp_bin)
    temp=[]
    for j in range(8-e_size):
        temp.append(0)
    exp_bin=temp+exp_bin
    if dec_num>=0:
        sign_bit=[0]
    else:
        sign_bit=[1]
    num_32_bit=sign_bit+["\t"]+exp_bin+["\t"]+mantissa
    print(i)
    print("Decimal number=",dec_num)
    print("IEEE 754 standard 32 bit number=",*num_32_bit)
```

```
alpha
Decimal number= 0.0072973525693
IEEE 754 standard 32 bit number= 0       0 1 1 1 0 1 1 1         1 1 0 1
1 1 1 0 0 0 1 1 1 1 0 1 0 1 0 0 0 0 1
c
Decimal number= 29979245800
IEEE 754 standard 32 bit number= 0       1 0 1 0 0 0 0 1         1 0 1 1
1 1 1 0 1 0 1 1 1 0 0 1 1 0 1 1 1 1 1
G
Decimal number= 6.6743e-08
IEEE 754 standard 32 bit number= 0       0 1 1 0 0 1 1 1         0 0 0 1
1 1 1 0 1 0 1 0 1 0 0 0 1 0 1 1 0 1 0
h
Decimal number= 4.1367e-15
IEEE 754 standard 32 bit number= 0       0 1 0 0 1 1 1 1         0 0 1 0
1 0 1 0 0 0 0 1 0 1 0 0 1 0 1 0 0 1 0
k
Decimal number= 8.6173e-05
IEEE 754 standard 32 bit number= 0       0 1 1 1 0 0 0 1         0 1 1 0
1 0 0 1 0 1 1 0 1 1 1 1 1 0 0 0 1 1 0
H0
Decimal number= 1.6202e-17
IEEE 754 standard 32 bit number= 0       0 1 0 0 0 1 1 1         0 0 1 0
1 0 1 0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0
M_sun
Decimal number= 3.955e+33
IEEE 754 standard 32 bit number= 0       1 1 1 0 1 1 1 0         1 0 0 0
0 1 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0
```

In [2]:

```
import numpy as np
dict_data={"alpha":0.0072973525693,"c":29979245800,"G":6.6743*10**(-8),"h
for i in dict_data.keys():
    dec_num=dict_data[i]
    #dec_num=6.6743*10**(-8)
    dec_l=int(np.floor(abs(dec_num)))
    dec_r=abs(dec_num)-dec_l
    def dec_l2b(num,l=[]):
        l.append(num%2)
        num=num//2
        if num!=0:
            dec_l2b(num,l)
        if num==0:
            l.reverse()
        return l
    def dec_r2b(num,l=[],count=0):
        d=num*2
        if d>=1:
            d1=1
            num=d-1
        else:
            d1=0
            num=d
        l.append(d1)
        count=count+1
        if count!=100:
            dec_r2b(num,l,count)
        return l
    a=dec_l2b(dec_l,[])
    b=dec_r2b(dec_r,[])
    c=a+b
    for j in range(len(c)):
        if c[j]==1:
            mantissa=c[j:]
            break;
    mantissa=mantissa[1:53]
    if dec_l!=0:
        exp=len(a)-1+1023
    else:
        exp=-j+1023
    exp_bin=dec_l2b(exp,[])
    e_size=len(exp_bin)
    temp=[]
    for j in range(11-e_size):
        temp.append(0)
    exp_bin=temp+exp_bin
    if dec_num>=0:
        sign_bit=[0]
    else:
        sign_bit=[1]
    num_32_bit=sign_bit+["\t"]+exp_bin+["\t"]+mantissa
    print(i)
    print("Decimal number=",dec_num)
    print("IEEE 754 standard 64 bit number=",*num_32_bit)
```

```
alpha
Decimal number= 0.0072973525693
IEEE 754 standard 64 bit number= 0       0 1 1 1 1 1 1 0 1 1 1   1 1 0 1
1 1 1 0 0 0 1 1 1 1 0 1 0 1 0 0 0 0 1 0 1 0 1 0 0 0 0 1 1 1 1 0 1 1 0 1 0
0 1 0 1 0 0 1 1 1 0 1
c
Decimal number= 29979245800
IEEE 754 standard 64 bit number= 0       1 0 0 0 0 1 0 0 0 0 1   1 0 1 1
1 1 1 0 1 0 1 1 1 0 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
G
Decimal number= 6.6743e-08
IEEE 754 standard 64 bit number= 0       0 1 1 1 1 1 0 0 1 1 1   0 0 0 1
1 1 1 0 1 0 1 0 1 0 0 0 1 0 1 1 0 1 0 0 0 1 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1
0 0 1 1 0 0 0 1 1 0 0
h
Decimal number= 4.1367e-15
IEEE 754 standard 64 bit number= 0       0 1 1 1 1 0 0 1 1 1 1   0 0 1 0
1 0 1 0 0 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1
1 1 1 0 1 0 1 0 1 0 1
k
Decimal number= 8.6173e-05
IEEE 754 standard 64 bit number= 0       0 1 1 1 1 1 1 0 0 0 1   0 1 1 0
1 0 0 1 0 1 1 0 1 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 1
0 0 1 0 1 1 1 0 1 0 1
H0
Decimal number= 1.6202e-17
IEEE 754 standard 64 bit number= 0       0 1 1 1 1 0 0 0 1 1 1   0 0 1 0
1 0 1 0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0
0 0 1
M_sun
Decimal number= 3.955e+33
IEEE 754 standard 64 bit number= 0       1 0 0 0 1 1 0 1 1 1 0   1 0 0 0
0 1 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 1 0 1 0 1 0 0 1 0 1 1 0
1 1 0 0 0 0 1 0 0 0 0
```

In [1]:

```
ieee_num_list=["010000001010100100110000000000000000000000000000000000000
for ieee_num in ieee_num_list:
        sign=(-1)**int(ieee_num[0])
        exp=ieee_num[1:12]
        mantissa=ieee_num[12:]
        def exp_bin2dec(exp):
                out=0
                for i in range(len(exp)):
                        out=out+int(exp[i])*(2**(11-i-1))
                out=out-1023
                return out
        def mant_bin2dec(mant):
                out=0
                for i in range(len(mant)):
                        out=out+int(mant[i])*(2**(-i-1))
                return out
        exp_dec=exp_bin2dec(exp)
        mantissa_dec=1+mant_bin2dec(mantissa)
        dec_num=sign*mantissa_dec*2**exp_dec
        print("ieee_num=",*ieee_num)
        print("dec_num=",dec_num)
```

```
ieee_num= 0 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
dec_num= 3224.0
ieee_num= 1 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
dec_num= -3224.0
ieee_num= 0 0 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
dec_num= 1.32421875
ieee_num= 0 0 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
dec_num= 1.3242187500000002
```

```
In [ ]:
```