大程 10report

运用 VS2017professional 和 c 语言代码如下：
代码如下，分别实现了浮点数到字符串，字符串到浮点数。浮点数的加减乘除。

```
#include<stdio.h>
#include<stdlib.h>

typedef unsigned int dwrd; //32 - bit
char* ftoa(dwrd num)
{
    int i = 0;
    char *str = (char *)malloc(sizeof(char) * 32);
    for (int i = 0; i < 32; i++)
    {
        if (num&(0x00000001 << i))str[i] = '1';
        else str[i] = '0';
    }
    return str;
}

dwrd atof(char* str)
{
    dwrd a=0x00000000;
    for (int i = 0; i < 32; i++)
    {
        if (str[i] == '1')
            a | (0x00000001 << i);

    }
    return a;
}

dwrd fadd(dwrd num1, dwrd num2)
{
    int temp1, temp2;
    int e;
    if (num1 & 0x80000000)
    {
        if (num2 & 0x80000000)
        {
            return -fadd(num1 & 0x7FFFFFFF, num2 & 0x7FFFFFFF);
        }
```

```
        else
        {
            return fsub(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
        }
    }
    else
    {
        if (num2 & 0x80000000)
        {
            return fsub(num1 & 0x7FFFFFFF, num2 & 0x7FFFFFFF);
        }
        else
        {
            temp1 = num1 >> 23;
            temp2 = num2 >> 23;
            if (temp1 > temp2)
            {
                e = temp1 - temp2;
                temp1 = num1&0x007FFFFF;
                temp2 = num2&0x007FFFFF;
                temp2 = temp2 >> e;
                temp1 = temp1 + temp2;
                temp1 = temp1 | (num1 & 0x7F800000);
                return temp1;
            }
            else
            {
                e = temp2 - temp1;
                temp2 = num1 & 0x007FFFFF;
                temp1 = num2 & 0x007FFFFF;
                temp1 = temp1 >> e;
                temp2 = temp2 + temp1;
                temp2 = temp2 | (num2 & 0x7F800000);
                return temp2;
            }
        }
    }
}

dwrd fsub(dwrd num1, dwrd num2)
{
    int temp1, temp2;
    int e;
    if (num1 & 0x80000000)
```

```
    {
        if (num2 & 0x80000000)
        {
            return fsub(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
        }
        else
        {
            return -fadd(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
        }
    }
    else
    {
        if (num2 & 0x80000000)
        {
            return fadd(num1 & 0x7FFFFFFF, num2 & 0x7FFFFFFF);
        }
        else
         {
             temp1 = num1 >> 23;
             temp2 = num2 >> 23;
             if (temp1 > temp2)
             {
                 e = temp1 - temp2;
                 temp1 = num1 & 0x007FFFFF;
                 temp2 = num2 & 0x007FFFFF;
                 temp2 = temp2 >> e;
                 temp1 = temp1 - temp2;
                 temp1 = temp1 | (num1 & 0x7F800000);
                 return temp1;
             }
             else
             {
                 e = temp2 - temp1;
                 temp2 = num1 & 0x007FFFFF;
                 temp1 = num2 & 0x007FFFFF;
                 temp1 = temp1 >> e;
                 temp2 = temp2 - temp1;
                 temp2 = temp2 | (num2 & 0x7F800000);
                 return temp2;
             }
         }
     }
}

```

```
127. dwrd fmul(dwrd num1, dwrd num2)
128. {
129.     int temp1, temp2;
130.     int e;
131.     if (num1 & 0x80000000)
132.     {
133.         if (num2 & 0x80000000)
134.         {
135.             return fmul(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
136.         }
137.         else
138.         {
139.             return -fmul(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
140.         }
141.     }
142.     else
143.     {
144.         if (num2 & 0x80000000)
145.         {
146.             return -fmul(num1 & 0x7FFFFFFF, num2 & 0x7FFFFFFF);
147.         }
148.         else
149.         {
150.             temp1 = num1 >> 23;
151.             temp2 = num2 >> 23;
152.             e = temp1 + temp2;
153.             temp1 = num1 & 0x007FFFFF;
154.             temp2 = num2 & 0x007FFFFF;
155.             temp1 = temp1 * temp2;
156.             temp1 = temp1 | (e << 23);
157.             return temp1;
158.         }
159.     }
160. }
161.
162. dwrd fdiv(dwrd num1, dwrd num2)
163. {
164.     int temp1, temp2;
165.     int e;
166.     if (num1 & 0x80000000)
167.     {
168.         if (num2 & 0x80000000)
169.         {
170.             return fdiv(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
```

```
171.         }
172.         else
173.         {
174.             return -fdiv(num2 & 0x7FFFFFFF, num1 & 0x7FFFFFFF);
175.         }
176.     }
177.     else
178.     {
179.         if (num2 & 0x80000000)
180.         {
181.             return -fdiv(num1 & 0x7FFFFFFF, num2 & 0x7FFFFFFF);
182.         }
183.         else
184.         {
185.             temp1 = num1 >> 23;
186.             temp2 = num2 >> 23;
187.             if (temp1 > temp2)
188.             {
189.                 e = temp1 - temp2;
190.                 temp1 = num1 & 0x007FFFFF;
191.                 temp2 = num2 & 0x007FFFFF;
192.                 temp2 = temp2 >> e;
193.                 temp1 = temp1 / temp2;
194.                 temp1 = temp1 | (e << 23);
195.                 return temp1;
196.             }
197.             else
198.             {
199.                 e = temp2 - temp1;
200.                 temp2 = num1 & 0x007FFFFF;
201.                 temp1 = num2 & 0x007FFFFF;
202.                 temp1 = temp1 >> e;
203.                 temp2 = temp2 / temp1;
204.                 temp2 = temp2 | (e << 23);
205.                 return temp2;
206.             }
207.         }
208.     }
209. }
```