俞楚凡 ICS 3.10 Lab

下面依次介绍本次Lab中我完成的三个实验。

- `expression_of_fpnumber.c`

  是构造浮点数并输出其十进制和二进制表示的过程。

  根据给定的四个数，分别计算得出其sign, exp, frac数据，并通过c语言内置的`union`联合体方法构造浮点数。

  具体运行结果如下：

  你可以输入一个浮点数，程序将会根据float和double的存储结构构造出这个浮点数并输出。正无穷大和NaN由手动构造给出。

```
D:\Files\Codes\FDU_SS_Courses\FDU_2025_ICS\3.10 lab>.\expression_of_fpnumber
20250309
float number: 20250308.000000
0 10010111 00110100111111101100010
double number: 20250308.000000
0 10000010111 0011010011111110110001000000000000000000000000000000
Positive infinity: 1.#INF00
0 11111111 00000000000000000000
NaN: 1.#QNAN0
0 11111111 11111111111111111111
```

```
D:\Files\Codes\FDU_SS_Courses\FDU_2025_ICS\3.10 lab>.\expression_of_fpnumber
-0.0625
float number: -0.062500
1 01111011 00000000000000000000
double number: -0.062500
1 01111111011 0000000000000000000000000000000000000000000000000000
Positive infinity: 1.#INF00
0 11111111 00000000000000000000
NaN: 1.#QNAN0
0 11111111 11111111111111111111
```

- `max_common_substring.c`

  实现了求解x, -x, (float)x, (double)x, 以及求解abs(x)与(float)x和(double)x的最长公共子串。

运行时样例：

```
D:\Files\Codes\FDU_SS_Courses\FDU_2025_ICS\3.10 lab>.\max_common_substring
20250309
x: 20250309
-x: -20250309
(float)x: 20250308.000000
(double)x: 20250309.000000
x: 00000001001101001111111011000101000000010011010011111111011000101
-x: 11111110110010110000000100111011111111110110010110000000100111011
float x: 0 10010111 00110100111111101100010
double x: 0 10000010111 0011010011111110110001010000000000000000000000000000
Longest common substring between |x| and (float)x: 2025030
Longest common substring between |x| and (double)x: 20250309
```

- `objdump_c.c`

实现了反汇编。

程序本体运行效果：

```
D:\Files\Codes\FDU_SS_Courses\FDU_2025_ICS\3.10 lab>.\objdump_c
1. int and int:
10 + 3 = 13
10 - 3 = 7
10 * 3 = 30
10 / 3 = 3

2. int and float:
5 + 2.5 = 7.50
5 - 2.5 = 2.50
5 * 2.5 = 12.50
5 / 2.5 = 2.00

3. float and double:
3.14 + 2.71828 = 5.858280
3.14 * 2.71828 = 8.535399

5. force transition:
7 / 2 = 3
(float)7 / 2 = 3.50
```

反汇编目标文件后得到下列汇编代码：

```
Disassembly of section .text:

0000000000000000 <main>:
   0:   55                      push   %rbp
   1:   48 89 e5                mov    %rsp,%rbp
   4:   48 83 ec 50             sub    $0x50,%rsp
   8:   e8 00 00 00 00          callq  d <main+0xd>
   d:   c7 45 fc 0a 00 00 00    movl   $0xa,-0x4(%rbp)
  14:   c7 45 f8 03 00 00 00    movl   $0x3,-0x8(%rbp)
  1b:   48 8d 0d 00 00 00 00    lea    0x0(%rip),%rcx
 # 22 <main+0x22>
```

```
 22:   e8 00 00 00 00          callq  27 <main+0x27>
 27:   8b 55 fc                mov    -0x4(%rbp),%edx
 2a:   8b 45 f8                mov    -0x8(%rbp),%eax
 2d:   8d 0c 02                lea    (%rdx,%rax,1),%ecx
 30:   8b 55 f8                mov    -0x8(%rbp),%edx
 33:   8b 45 fc                mov    -0x4(%rbp),%eax
 36:   41 89 c9                mov    %ecx,%r9d
 39:   41 89 d0                mov    %edx,%r8d
 3c:   89 c2                   mov    %eax,%edx
 3e:   48 8d 0d 10 00 00 00    lea    0x10(%rip),%rcx
# 55 <main+0x55>
 45:   e8 00 00 00 00          callq  4a <main+0x4a>
 4a:   8b 45 fc                mov    -0x4(%rbp),%eax
 4d:   2b 45 f8                sub    -0x8(%rbp),%eax
 50:   89 c2                   mov    %eax,%edx
 52:   8b 4d f8                mov    -0x8(%rbp),%ecx
 55:   8b 45 fc                mov    -0x4(%rbp),%eax
 58:   41 89 d1                mov    %edx,%r9d
 5b:   41 89 c8                mov    %ecx,%r8d
 5e:   89 c2                   mov    %eax,%edx
 60:   48 8d 0d 1e 00 00 00    lea    0x1e(%rip),%rcx
# 85 <main+0x85>
 67:   e8 00 00 00 00          callq  6c <main+0x6c>
 6c:   8b 45 fc                mov    -0x4(%rbp),%eax
 6f:   0f af 45 f8             imul   -0x8(%rbp),%eax
 73:   89 c2                   mov    %eax,%edx
 75:   8b 4d f8                mov    -0x8(%rbp),%ecx
 78:   8b 45 fc                mov    -0x4(%rbp),%eax
 7b:   41 89 d1                mov    %edx,%r9d
 7e:   41 89 c8                mov    %ecx,%r8d
 81:   89 c2                   mov    %eax,%edx
 83:   48 8d 0d 2c 00 00 00    lea    0x2c(%rip),%rcx
# b6 <main+0xb6>
 8a:   e8 00 00 00 00          callq  8f <main+0x8f>
 8f:   8b 45 fc                mov    -0x4(%rbp),%eax
 92:   99                      cltd
 93:   f7 7d f8                idivl  -0x8(%rbp)
 96:   89 c1                   mov    %eax,%ecx
 98:   8b 55 f8                mov    -0x8(%rbp),%edx
 9b:   8b 45 fc                mov    -0x4(%rbp),%eax
 9e:   41 89 c9                mov    %ecx,%r9d
 a1:   41 89 d0                mov    %edx,%r8d
```

```
 a4:   89 c2                   mov    %eax,%edx
 a6:   48 8d 0d 3a 00 00 00    lea    0x3a(%rip),%rcx
# e7 <main+0xe7>
 ad:   e8 00 00 00 00          callq  b2 <main+0xb2>
 b2:   c7 45 f4 05 00 00 00    movl   $0x5,-0xc(%rbp)
 b9:   f3 0f 10 05 20 01 00    movss  0x120(%rip),%xmm0
  # 1e1 <main+0x1e1>
 c0:   00
 c1:   f3 0f 11 45 f0          movss  %xmm0,-0x10(%rbp)
 c6:   48 8d 0d 49 00 00 00    lea    0x49(%rip),%rcx
# 116 <main+0x116>
 cd:   e8 00 00 00 00          callq  d2 <main+0xd2>
 d2:   f3 0f 2a 45 f4          cvtsi2ssl -0xc(%rbp),%xmm0
 d7:   f3 0f 58 45 f0          addss  -0x10(%rbp),%xmm0
 dc:   f3 0f 5a c8             cvtss2sd %xmm0,%xmm1
 e0:   f3 0f 5a 45 f0          cvtss2sd -0x10(%rbp),%xmm0
 e5:   66 48 0f 7e c8          movq   %xmm1,%rax
 ea:   48 89 c2                mov    %rax,%rdx
 ed:   48 89 d1                mov    %rdx,%rcx
 f0:   66 48 0f 6e c8          movq   %rax,%xmm1
 f5:   66 48 0f 7e c0          movq   %xmm0,%rax
 fa:   48 89 c2                mov    %rax,%rdx
 fd:   66 48 0f 6e c0          movq   %rax,%xmm0
102:   8b 45 f4                mov    -0xc(%rbp),%eax
105:   66 48 0f 6e d9          movq   %rcx,%xmm3
10a:   66 49 0f 7e c9          movq   %xmm1,%r9
10f:   66 48 0f 6e d2          movq   %rdx,%xmm2
114:   66 49 0f 7e c0          movq   %xmm0,%r8
119:   89 c2                   mov    %eax,%edx
11b:   48 8d 0d 5b 00 00 00    lea    0x5b(%rip),%rcx
# 17d <main+0x17d>
122:   e8 00 00 00 00          callq  127 <main+0x127>
127:   f3 0f 2a 45 f4          cvtsi2ssl -0xc(%rbp),%xmm0
12c:   f3 0f 5c 45 f0          subss  -0x10(%rbp),%xmm0
131:   f3 0f 5a c8             cvtss2sd %xmm0,%xmm1
135:   f3 0f 5a 45 f0          cvtss2sd -0x10(%rbp),%xmm0
13a:   66 48 0f 7e c8          movq   %xmm1,%rax
13f:   48 89 c2                mov    %rax,%rdx
142:   48 89 d1                mov    %rdx,%rcx
145:   66 48 0f 6e c8          movq   %rax,%xmm1
14a:   66 48 0f 7e c0          movq   %xmm0,%rax
14f:   48 89 c2                mov    %rax,%rdx
```

```
152:   66 48 0f 6e c0          movq   %rax,%xmm0
157:   8b 45 f4                mov    -0xc(%rbp),%eax
15a:   66 48 0f 6e d9          movq   %rcx,%xmm3
15f:   66 49 0f 7e c9          movq   %xmm1,%r9
164:   66 48 0f 6e d2          movq   %rdx,%xmm2
169:   66 49 0f 7e c0          movq   %xmm0,%r8
16e:   89 c2                   mov    %eax,%edx
170:   48 8d 0d 6d 00 00 00    lea    0x6d(%rip),%rcx
# 1e4 <main+0x1e4>
177:   e8 00 00 00 00          callq  17c <main+0x17c>
17c:   f3 0f 2a 45 f4          cvtsi2ssl -0xc(%rbp),%xmm0
181:   f3 0f 59 45 f0          mulss  -0x10(%rbp),%xmm0
186:   f3 0f 5a c8             cvtss2sd %xmm0,%xmm1
18a:   f3 0f 5a 45 f0          cvtss2sd -0x10(%rbp),%xmm0
18f:   66 48 0f 7e c8          movq   %xmm1,%rax
194:   48 89 c2                mov    %rax,%rdx
197:   48 89 d1                mov    %rdx,%rcx
19a:   66 48 0f 6e c8          movq   %rax,%xmm1
19f:   66 48 0f 7e c0          movq   %xmm0,%rax
1a4:   48 89 c2                mov    %rax,%rdx
1a7:   66 48 0f 6e c0          movq   %rax,%xmm0
1ac:   8b 45 f4                mov    -0xc(%rbp),%eax
1af:   66 48 0f 6e d9          movq   %rcx,%xmm3
1b4:   66 49 0f 7e c9          movq   %xmm1,%r9
1b9:   66 48 0f 6e d2          movq   %rdx,%xmm2
1be:   66 49 0f 7e c0          movq   %xmm0,%r8
1c3:   89 c2                   mov    %eax,%edx
1c5:   48 8d 0d 7f 00 00 00    lea    0x7f(%rip),%rcx
# 24b <main+0x24b>
1cc:   e8 00 00 00 00          callq  1d1 <main+0x1d1>
1d1:   f3 0f 2a 45 f4          cvtsi2ssl -0xc(%rbp),%xmm0
1d6:   f3 0f 5e 45 f0          divss  -0x10(%rbp),%xmm0
1db:   f3 0f 5a c8             cvtss2sd %xmm0,%xmm1
1df:   f3 0f 5a 45 f0          cvtss2sd -0x10(%rbp),%xmm0
1e4:   66 48 0f 7e c8          movq   %xmm1,%rax
1e9:   48 89 c2                mov    %rax,%rdx
1ec:   48 89 d1                mov    %rdx,%rcx
1ef:   66 48 0f 6e c8          movq   %rax,%xmm1
1f4:   66 48 0f 7e c0          movq   %xmm0,%rax
1f9:   48 89 c2                mov    %rax,%rdx
1fc:   66 48 0f 6e c0          movq   %rax,%xmm0
201:   8b 45 f4                mov    -0xc(%rbp),%eax
```

```
204:   66 48 0f 6e d9          movq   %rcx,%xmm3
209:   66 49 0f 7e c9          movq   %xmm1,%r9
20e:   66 48 0f 6e d2          movq   %rdx,%xmm2
213:   66 49 0f 7e c0          movq   %xmm0,%r8
218:   89 c2                   mov    %eax,%edx
21a:   48 8d 0d 91 00 00 00    lea    0x91(%rip),%rcx
# 2b2 <main+0x2b2>
221:   e8 00 00 00 00          callq  226 <main+0x226>
226:   f3 0f 10 05 24 01 00    movss  0x124(%rip),%xmm0
  # 352 <main+0x352>
22d:   00
22e:   f3 0f 11 45 ec          movss  %xmm0,-0x14(%rbp)
233:   f2 0f 10 05 28 01 00    movsd  0x128(%rip),%xmm0
  # 363 <main+0x363>
23a:   00
23b:   f2 0f 11 45 e0          movsd  %xmm0,-0x20(%rbp)
240:   48 8d 0d a4 00 00 00    lea    0xa4(%rip),%rcx
# 2eb <main+0x2eb>
247:   e8 00 00 00 00          callq  24c <main+0x24c>
24c:   f3 0f 5a 45 ec          cvtss2sd -0x14(%rbp),%xmm0
251:   f2 0f 58 45 e0          addsd  -0x20(%rbp),%xmm0
256:   f3 0f 5a 55 ec          cvtss2sd -0x14(%rbp),%xmm2
25b:   66 48 0f 7e c0          movq   %xmm0,%rax
260:   48 89 c2                mov    %rax,%rdx
263:   48 89 d1                mov    %rdx,%rcx
266:   66 48 0f 6e e0          movq   %rax,%xmm4
26b:   f2 0f 10 4d e0          movsd  -0x20(%rbp),%xmm1
270:   f2 0f 10 45 e0          movsd  -0x20(%rbp),%xmm0
275:   66 48 0f 7e d0          movq   %xmm2,%rax
27a:   48 89 c2                mov    %rax,%rdx
27d:   66 48 0f 6e d9          movq   %rcx,%xmm3
282:   66 49 0f 7e e1          movq   %xmm4,%r9
287:   66 0f 28 d1             movapd %xmm1,%xmm2
28b:   66 49 0f 7e c0          movq   %xmm0,%r8
290:   66 48 0f 6e ca          movq   %rdx,%xmm1
295:   48 89 c2                mov    %rax,%rdx
298:   48 8d 0d b9 00 00 00    lea    0xb9(%rip),%rcx
# 358 <main+0x358>
29f:   e8 00 00 00 00          callq  2a4 <main+0x2a4>
2a4:   f3 0f 5a 45 ec          cvtss2sd -0x14(%rbp),%xmm0
2a9:   f2 0f 59 45 e0          mulsd  -0x20(%rbp),%xmm0
2ae:   f3 0f 5a 55 ec          cvtss2sd -0x14(%rbp),%xmm2
```

```
2b3:   66 48 0f 7e c0          movq   %xmm0,%rax
2b8:   48 89 c2                mov    %rax,%rdx
2bb:   48 89 d1                mov    %rdx,%rcx
2be:   66 48 0f 6e e0          movq   %rax,%xmm4
2c3:   f2 0f 10 4d e0          movsd  -0x20(%rbp),%xmm1
2c8:   f2 0f 10 45 e0          movsd  -0x20(%rbp),%xmm0
2cd:   66 48 0f 7e d0          movq   %xmm2,%rax
2d2:   48 89 c2                mov    %rax,%rdx
2d5:   66 48 0f 6e d9          movq   %rcx,%xmm3
2da:   66 49 0f 7e e1          movq   %xmm4,%r9
2df:   66 0f 28 d1             movapd %xmm1,%xmm2
2e3:   66 49 0f 7e c0          movq   %xmm0,%r8
2e8:   66 48 0f 6e ca          movq   %rdx,%xmm1
2ed:   48 89 c2                mov    %rax,%rdx
2f0:   48 8d 0d cd 00 00 00    lea    0xcd(%rip),%rcx
# 3c4 <main+0x3c4>
2f7:   e8 00 00 00 00          callq  2fc <main+0x2fc>
2fc:   c7 45 dc 07 00 00 00    movl   $0x7,-0x24(%rbp)
303:   c7 45 d8 02 00 00 00    movl   $0x2,-0x28(%rbp)
30a:   48 8d 0d e2 00 00 00    lea    0xe2(%rip),%rcx
# 3f3 <main+0x3f3>
311:   e8 00 00 00 00          callq  316 <main+0x316>
316:   8b 45 dc                mov    -0x24(%rbp),%eax
319:   99                      cltd
31a:   f7 7d d8                idivl  -0x28(%rbp)
31d:   89 c1                   mov    %eax,%ecx
31f:   8b 55 d8                mov    -0x28(%rbp),%edx
322:   8b 45 dc                mov    -0x24(%rbp),%eax
325:   41 89 c9                mov    %ecx,%r9d
328:   41 89 d0                mov    %edx,%r8d
32b:   89 c2                   mov    %eax,%edx
32d:   48 8d 0d f8 00 00 00    lea    0xf8(%rip),%rcx
# 42c <main+0x42c>
334:   e8 00 00 00 00          callq  339 <main+0x339>
339:   f3 0f 2a 45 dc          cvtsi2ssl -0x24(%rbp),%xmm0
33e:   f3 0f 2a 4d d8          cvtsi2ssl -0x28(%rbp),%xmm1
343:   f3 0f 5e c1             divss  %xmm1,%xmm0
347:   f3 0f 5a c0             cvtss2sd %xmm0,%xmm0
34b:   66 48 0f 7e c0          movq   %xmm0,%rax
350:   48 89 c2                mov    %rax,%rdx
353:   48 89 d1                mov    %rdx,%rcx
356:   66 48 0f 6e c0          movq   %rax,%xmm0
```

```
 35b:   8b 55 d8                mov    -0x28(%rbp),%edx
 35e:   8b 45 dc                mov    -0x24(%rbp),%eax
 361:   66 48 0f 6e d9          movq   %rcx,%xmm3
 366:   66 49 0f 7e c1          movq   %xmm0,%r9
 36b:   41 89 d0                mov    %edx,%r8d
 36e:   89 c2                   mov    %eax,%edx
 370:   48 8d 0d 06 01 00 00    lea    0x106(%rip),%rcx
   # 47d <main+0x47d>
 377:   e8 00 00 00 00          callq  37c <main+0x37c>
 37c:   b8 00 00 00 00          mov    $0x0,%eax
 381:   48 83 c4 50             add    $0x50,%rsp
 385:   5d                      pop    %rbp
 386:   c3                      retq
 387:   90                      nop
 388:   90                      nop
 389:   90                      nop
 38a:   90                      nop
 38b:   90                      nop
 38c:   90                      nop
 38d:   90                      nop
 38e:   90                      nop
 38f:   90                      nop
```

直接编译得到的.s文件如下：

```
    .file   "objdump_c.c"
    .text
    .def    __main; .scl    2;  .type   32; .endef
    .section .rdata,"dr"
.LC0:
    .ascii "1. int and int:\0"
.LC1:
    .ascii "%d + %d = %d\12\0"
.LC2:
    .ascii "%d - %d = %d\12\0"
.LC3:
    .ascii "%d * %d = %d\12\0"
.LC4:
    .ascii "%d / %d = %d\12\12\0"
.LC6:
    .ascii "2. int and float:\0"
.LC7:
```

```
    .ascii "%d + %.1f = %.2f\12\0"
.LC8:
    .ascii "%d - %.1f = %.2f\12\0"
.LC9:
    .ascii "%d * %.1f = %.2f\12\0"
.LC10:
    .ascii "%d / %.1f = %.2f\12\12\0"
.LC13:
    .ascii "3. float and double:\0"
.LC14:
    .ascii "%.2f + %.5f = %.6f\12\0"
.LC15:
    .ascii "%.2f * %.5f = %.6f\12\12\0"
.LC16:
    .ascii "5. force transition: \0"
.LC17:
    .ascii "%d / %d = %d\12\0"
.LC18:
    .ascii "(float)%d / %d = %.2f\12\0"
    .text
    .globl  main
    .def    main;   .scl    2;  .type   32; .endef
    .seh_proc   main
main:
    pushq   %rbp
    .seh_pushreg    %rbp
    movq    %rsp, %rbp
    .seh_setframe   %rbp, 0
    subq    $80, %rsp
    .seh_stackalloc 80
    .seh_endprologue
    call    __main
    movl    $10, -4(%rbp)
    movl    $3, -8(%rbp)
    leaq    .LC0(%rip), %rcx
    call    puts
    movl    -4(%rbp), %edx
    movl    -8(%rbp), %eax
    leal    (%rdx,%rax), %ecx
    movl    -8(%rbp), %edx
    movl    -4(%rbp), %eax
    movl    %ecx, %r9d
```

```
    movl    %edx, %r8d
    movl    %eax, %edx
    leaq    .LC1(%rip), %rcx
    call    printf
    movl    -4(%rbp), %eax
    subl    -8(%rbp), %eax
    movl    %eax, %edx
    movl    -8(%rbp), %ecx
    movl    -4(%rbp), %eax
    movl    %edx, %r9d
    movl    %ecx, %r8d
    movl    %eax, %edx
    leaq    .LC2(%rip), %rcx
    call    printf
    movl    -4(%rbp), %eax
    imull   -8(%rbp), %eax
    movl    %eax, %edx
    movl    -8(%rbp), %ecx
    movl    -4(%rbp), %eax
    movl    %edx, %r9d
    movl    %ecx, %r8d
    movl    %eax, %edx
    leaq    .LC3(%rip), %rcx
    call    printf
    movl    -4(%rbp), %eax
    cltd
    idivl   -8(%rbp)
    movl    %eax, %ecx
    movl    -8(%rbp), %edx
    movl    -4(%rbp), %eax
    movl    %ecx, %r9d
    movl    %edx, %r8d
    movl    %eax, %edx
    leaq    .LC4(%rip), %rcx
    call    printf
    movl    $5, -12(%rbp)
    movss   .LC5(%rip), %xmm0
    movss   %xmm0, -16(%rbp)
    leaq    .LC6(%rip), %rcx
    call    puts
    cvtsi2ss    -12(%rbp), %xmm0
    addss   -16(%rbp), %xmm0
```

```
    cvtss2sd    %xmm0, %xmm1
    cvtss2sd    -16(%rbp), %xmm0
    movq    %xmm1, %rax
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm1
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rax, %xmm0
    movl    -12(%rbp), %eax
    movq    %rcx, %xmm3
    movq    %xmm1, %r9
    movq    %rdx, %xmm2
    movq    %xmm0, %r8
    movl    %eax, %edx
    leaq    .LC7(%rip), %rcx
    call    printf
    cvtsi2ss    -12(%rbp), %xmm0
    subss   -16(%rbp), %xmm0
    cvtss2sd    %xmm0, %xmm1
    cvtss2sd    -16(%rbp), %xmm0
    movq    %xmm1, %rax
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm1
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rax, %xmm0
    movl    -12(%rbp), %eax
    movq    %rcx, %xmm3
    movq    %xmm1, %r9
    movq    %rdx, %xmm2
    movq    %xmm0, %r8
    movl    %eax, %edx
    leaq    .LC8(%rip), %rcx
    call    printf
    cvtsi2ss    -12(%rbp), %xmm0
    mulss   -16(%rbp), %xmm0
    cvtss2sd    %xmm0, %xmm1
    cvtss2sd    -16(%rbp), %xmm0
    movq    %xmm1, %rax
    movq    %rax, %rdx
```

```
    movq    %rdx, %rcx
    movq    %rax, %xmm1
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rax, %xmm0
    movl    -12(%rbp), %eax
    movq    %rcx, %xmm3
    movq    %xmm1, %r9
    movq    %rdx, %xmm2
    movq    %xmm0, %r8
    movl    %eax, %edx
    leaq    .LC9(%rip), %rcx
    call    printf
    cvtsi2ss    -12(%rbp), %xmm0
    divss   -16(%rbp), %xmm0
    cvtss2sd    %xmm0, %xmm1
    cvtss2sd    -16(%rbp), %xmm0
    movq    %xmm1, %rax
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm1
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rax, %xmm0
    movl    -12(%rbp), %eax
    movq    %rcx, %xmm3
    movq    %xmm1, %r9
    movq    %rdx, %xmm2
    movq    %xmm0, %r8
    movl    %eax, %edx
    leaq    .LC10(%rip), %rcx
    call    printf
    movss   .LC11(%rip), %xmm0
    movss   %xmm0, -20(%rbp)
    movsd   .LC12(%rip), %xmm0
    movsd   %xmm0, -32(%rbp)
    leaq    .LC13(%rip), %rcx
    call    puts
    cvtss2sd    -20(%rbp), %xmm0
    addsd   -32(%rbp), %xmm0
    cvtss2sd    -20(%rbp), %xmm2
    movq    %xmm0, %rax
```

```
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm4
    movsd   -32(%rbp), %xmm1
    movsd   -32(%rbp), %xmm0
    movq    %xmm2, %rax
    movq    %rax, %rdx
    movq    %rcx, %xmm3
    movq    %xmm4, %r9
    movapd  %xmm1, %xmm2
    movq    %xmm0, %r8
    movq    %rdx, %xmm1
    movq    %rax, %rdx
    leaq    .LC14(%rip), %rcx
    call    printf
    cvtss2sd    -20(%rbp), %xmm0
    mulsd   -32(%rbp), %xmm0
    cvtss2sd    -20(%rbp), %xmm2
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm4
    movsd   -32(%rbp), %xmm1
    movsd   -32(%rbp), %xmm0
    movq    %xmm2, %rax
    movq    %rax, %rdx
    movq    %rcx, %xmm3
    movq    %xmm4, %r9
    movapd  %xmm1, %xmm2
    movq    %xmm0, %r8
    movq    %rdx, %xmm1
    movq    %rax, %rdx
    leaq    .LC15(%rip), %rcx
    call    printf
    movl    $7, -36(%rbp)
    movl    $2, -40(%rbp)
    leaq    .LC16(%rip), %rcx
    call    puts
    movl    -36(%rbp), %eax
    cltd
    idivl   -40(%rbp)
    movl    %eax, %ecx
```

```
    movl    -40(%rbp), %edx
    movl    -36(%rbp), %eax
    movl    %ecx, %r9d
    movl    %edx, %r8d
    movl    %eax, %edx
    leaq    .LC17(%rip), %rcx
    call    printf
    cvtsi2ss    -36(%rbp), %xmm0
    cvtsi2ss    -40(%rbp), %xmm1
    divss   %xmm1, %xmm0
    cvtss2sd    %xmm0, %xmm0
    movq    %xmm0, %rax
    movq    %rax, %rdx
    movq    %rdx, %rcx
    movq    %rax, %xmm0
    movl    -40(%rbp), %edx
    movl    -36(%rbp), %eax
    movq    %rcx, %xmm3
    movq    %xmm0, %r9
    movl    %edx, %r8d
    movl    %eax, %edx
    leaq    .LC18(%rip), %rcx
    call    printf
    movl    $0, %eax
    addq    $80, %rsp
    popq    %rbp
    ret
    .seh_endproc
    .section .rdata,"dr"
    .align 4
.LC5:
    .long   1075838976
    .align 4
.LC11:
    .long   1078523331
    .align 8
.LC12:
    .long   -1783957616
    .long   1074118409
    .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64
project) 8.1.0"
    .def    puts;   .scl    2;  .type   32; .endef
```

```
    .def     printf; .scl    2;  .type   32; .endef
```

感觉上区别还蛮大的。

涉及到不同数据类型对应的运算和数据类型转换对应的指令，我观察到以下这些：

a.

- `ADD`：加法操作。
- `ADC`：带进位的加法。
- `SUB`：减法操作。
- `SBB`：带借位的减法。
- `MUL`：无符号数乘法。
- `IMUL`：有符号数乘法。
- `DIV`：无符号数除法。
- `IDIV`：有符号数除法。
- `AND`：按位与。
- `OR`：按位或。
- `XOR`：按位异或。
- `NOT`：按位非。