学号：2022141410193

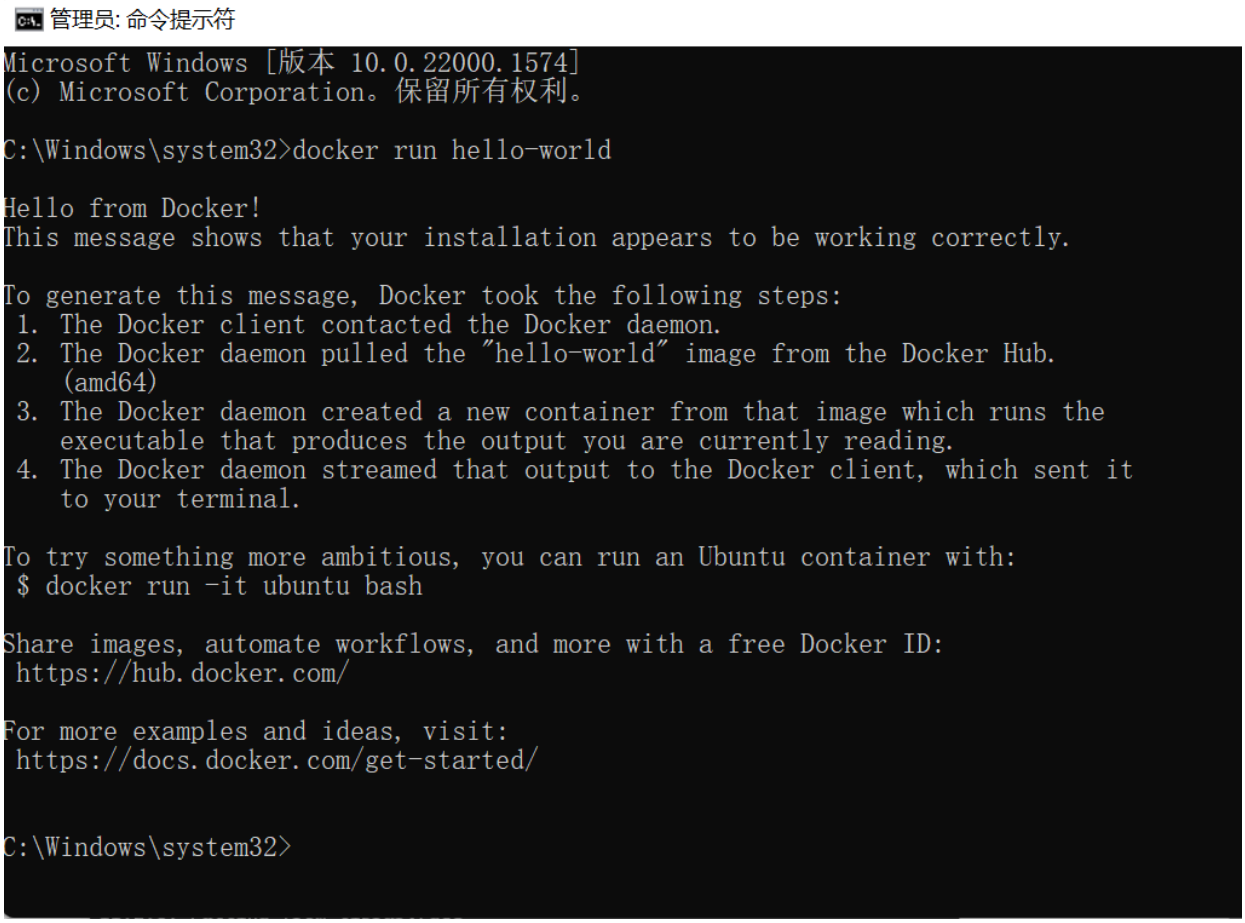
姓名：王枭越

1 作业内容概述

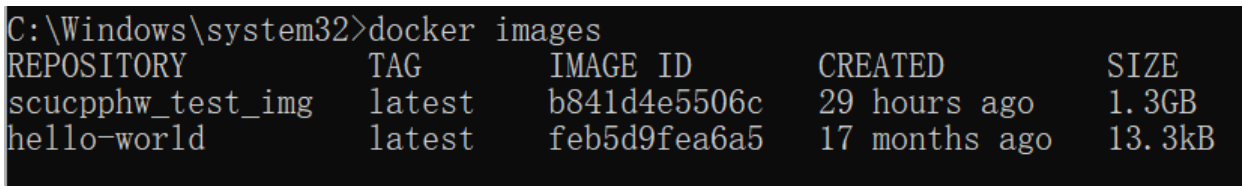
无

2 配置对应的docker 环境

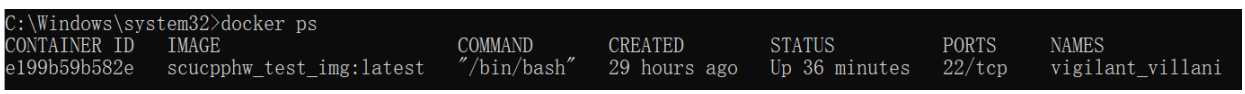
1. docker run hello-world



2. docker images



3. docker ps



4. 使用docker exec -it /bin/bash 进入docker 后cd /ws/code/的截图

```
C:\Windows\system32>docker exec -it e199b59b582e /bin/bash
root@e199b59b582e:/# cd /ws/code
root@e199b59b582e:/ws/code#
```

3 利用g++ 编译单文件

- 单步

1. 单步生成a.out可执行文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ test.cpp
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ls
a.out  inefficency.cpp  other_test  test.cpp
root@e199b59b582e:/ws/code/Assignment_1_test/question3#
```

2. 执行a.out文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ./a.out
vector v after call to generate_n() with lambda: 1 1 2 3 5 8 13 21 34
x: 1 y: 1
vector v after 1st call to fillVector(): 1 2 3 4 5 6 7 8 9
vector v after 2nd call to fillVector(): 10 11 12 13 14 15 16 17 18
```

- 多步

1. 预处理Pre-processing，生成.i 文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ -E test.cpp -o test.i
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ls
a.out  inefficency.cpp  other_test  test.cpp  test.i
```

2. 编译-Compiling，生成.s 文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ -S test.i -o test.s
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ls
a.out  inefficency.cpp  other_test  test.cpp  test.i  test.s
```

3. 汇编-Assembing，生成.o 文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ -c test.s -o test.o
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ls
a.out  inefficency.cpp  other_test  test.cpp  test.i  test.o  test.s
```

4. 链接-Lingking，生成bin 二进制文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ test.o -o test
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ls
a.out  inefficency.cpp  other_test  test  test.cpp  test.i  test.o  test.s
```

5. 运行最后的test可执行文件

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# ./test
vector v after call to generate_n() with lambda: 1 1 2 3 5 8 13 21 34
x: 1 y: 1
vector v after 1st call to fillVector(): 1 2 3 4 5 6 7 8 9
vector v after 2nd call to fillVector(): 10 11 12 13 14 15 16 17 18
```

- 利用time 命令打印inefficency.cpp 优化编译与非优化编译的执行信息

1. 未优化编译

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ inefficency.cpp -o without_o.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3# sudo time ./without_o.out
sudo: time: command not found
root@e199b59b582e:/ws/code/Assignment_1_test/question3# chmod +x without_o.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3# time ./without_o.out
result = 100904034

real    0m2.266s
user    0m2.253s
sys     0m0.010s
```

2. 优化编译

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3# g++ inefficency.cpp -O2 -o  with_o.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3# chmod +x with_o.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3# time ./with_o.out
result = 100904034

real    0m0.006s
user    0m0.004s
sys     0m0.000s
```

- 选择3 到4 个其他的test 文件，尝试利用不同的g++ 参数进行编译，给出对应生成文件与最终的执行截图

1. 使用c++17 标准编译test_class.cpp

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ls
test_class.cpp  test_class_size.cpp  test_default_parameter.cpp  test_move.cpp  test_noexcept.cpp  test_ptr.cpp  test_raii.cpp
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# g++ -std=c++17 test_class.cpp -o test_class.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ls
test_class.cpp  test_class.out  test_class_size.cpp  test_default_parameter.cpp  test_move.cpp  test_noexcept.cpp  test_ptr.cpp  test_raii.cpp
```

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ./test_class.out
1       #21325302 is created
1       #58320212 is created
5       #21325302       5000    5000
25      #58320212       10000   10000
45      #21325302       5500    10500
60      #58320212       -4000   6000
90      #21325302       27.64   10527.6
90      #58320212       21.78   6021.78
#21325302       Balance: 10527.6
#58320212       Balance: 6021.78
```

2. 打印出g++ 提供的警告信息

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# g++ -Wall test_class_size.cpp -o test_class_size.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# chmod +x test_class_size.out
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ./test_class_size.out
Size of Student: 8
Number:    12345678
Level:     sophomore
Grade:     B
```

因为没错所以并没有打印警告信息

3. 产生带调试信息的可执行文件test_default_parameter

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# g++ -g test_default_parameter.cpp -o test_default_parameter
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ls
test_class.cpp  test_class_size.cpp  test_default_parameter      test_move.cpp      test_ptr.cpp
test_class.out  test_class_size.out  test_default_parameter.cpp  test_noexcept.cpp  test_raii.cpp
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# ./test_default_parameter
Some box data is    10   12   15        1800
Some box data is    10   12    3        360
Some box data is    10    2    3        60
```

```
root@e199b59b582e:/ws/code/Assignment_1_test/question3/other_test# readelf -S test_default_parameter | grep -i debug
  [26] .debug_aranges    PROGBITS         0000000000000000  00003082
  [27] .debug_info       PROGBITS         0000000000000000  000030c2
  [28] .debug_abbrev     PROGBITS         0000000000000000  000057f0
  [29] .debug_line       PROGBITS         0000000000000000  00005de9
  [30] .debug_str        PROGBITS         0000000000000000  00005faf
  [31] .debug_rnglists   PROGBITS         0000000000000000  000072bb
  [32] .debug_line_str   PROGBITS         0000000000000000  000072dd
```

4 利用GDB 调试文件

1. 对于代码片段一，分别追踪前后两次调用函数sumOfSquare() 时函数调用的栈帧与层级关系，并给出过程的相关截图

```
root@e199b59b582e:/ws/code/Assignment_1_test/question4# gdb test_function_overload
GNU gdb (Debian 10.1-1.7) 10.1.90.20210103-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.
```

```
--Type <RET> for more, q to quit, c to continue without paging--

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from test_function_overload...
(gdb) b 16
Breakpoint 1 at 0x401241: file test_function_overload.cpp, line 16.
(gdb) b 21
Breakpoint 2 at 0x4012ab: file test_function_overload.cpp, line 21.
(gdb) r
Starting program: /ws/code/Assignment_1_test/question4/test_function_overload
warning: Error disabling address space randomization: Operation not permitted
warning: File "/usr/local/lib64/libstdc++.so.6.0.29-gdb.py" auto-loading has been declined by your `auto-load safe-path' set to "$debugdir:$datadir/auto-load".
```

```
To enable execution of this file add
        add-auto-load-safe-path /usr/local/lib64/libstdc++.so.6.0.29-gdb.py
line to your configuration file "/root/.gdbinit".
To completely disable this security protection add
        set auto-load safe-path /
line to your configuration file "/root/.gdbinit".
For more information about this security protection see the
"Auto-loading safe path" section in the GDB manual.  E.g., run from the shell:
        info "(gdb)Auto-loading safe path"
Enter two integer: 3 4

Breakpoint 1, main () at test_function_overload.cpp:16
--Type <RET> for more, q to quit, c to continue without paging--
```

```
16              cout << "Their sum of square: " << sumOfSquare(m, n) << endl;
(gdb) s
sumOfSquare (a=3, b=4) at test_function_overload.cpp:5
5               return a * a + b * b;
(gdb) bt
#0  sumOfSquare (a=3, b=4) at test_function_overload.cpp:5
#1  0x0000000000401262 in main () at test_function_overload.cpp:16
(gdb) c
Continuing.
Their sum of square: 25
Enter two real number: 11.4 51.4

Breakpoint 2, main () at test_function_overload.cpp:21
```

```
21              cout << "Their sum of square: " << sumOfSquare(x, y) << endl;
(gdb) s
sumOfSquare (a=11.4, b=51.399999999999999) at test_function_overload.cpp:9
9               return a * a + b * b;
(gdb) bt
#0  sumOfSquare (a=11.4, b=51.399999999999999) at test_function_overload.cpp:9
#1  0x00000000004012d4 in main () at test_function_overload.cpp:21
(gdb) c
Continuing.
Their sum of square: 2771.92
[Inferior 1 (process 8588) exited normally]
(gdb) n
The program is not being run.
```

```
The program is not being run.
(gdb) q
root@e199b59b582e:/ws/code/Assignment_1_test/question4#
```

2. 对于代码片段二，我们希望您能够追踪每次循环的变量y 的具体变量值，并思考最后打印的j 数值的意义

```
root@e199b59b582e:/ws/code/Assignment_1_test/question4# gdb test_range_based
GNU gdb (Debian 10.1-1.7) 10.1.90.20210103-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.
--Type <RET> for more, q to quit, c to continue without paging--

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from test_range_based...
(gdb) b 15
Breakpoint 1 at 0x4012e1: file test_range_based.cpp, line 15.
(gdb) b 20
Breakpoint 2 at 0x40134b: file test_range_based.cpp, line 20.
(gdb) b 25
Breakpoint 3 at 0x4013b7: file test_range_based.cpp, line 25.
(gdb) b 30
Breakpoint 4 at 0x401423: file test_range_based.cpp, line 30.
(gdb) r
Starting program: /ws/code/Assignment_1_test/question4/test_range_based
warning: Error disabling address space randomization: Operation not permitted
warning: File "/usr/local/lib64/libstdc++.so.6.0.29-gdb.py" auto-loading has beer
To enable execution of this file add
        add-auto-load-safe-path /usr/local/lib64/libstdc++.so.6.0.29-gdb.py
line to your configuration file "/root/.gdbinit".
To completely disable this security protection add
        set auto-load safe-path /
line to your configuration file "/root/.gdbinit".
For more information about this security protection see the
"Auto-loading safe path" section in the GDB manual.  E.g., run from the shell:
        info "(gdb)Auto-loading safe path"

Breakpoint 1, main () at test_range_based.cpp:15
--Type <RET> for more, q to quit, c to continue without paging--
15          cout << endl;
(gdb) c
Continuing.
1 2 3 4 5 6 7 8 9 10
end of integer array test

0.14159 1.14159 2.14159 3.14159 4.14159 5.14159 6.14159 7.14159 8.14159 9.14159
end of vector test
[Inferior 1 (process 22321) exited normally]
(gdb) q
root@e199b59b582e:/ws/code/Assignment_1_test/question4#
```

本可以用打点和

```
$(gdb) display y
```

追踪y的值，但是本身是打印语句，只要每次遍历打印完设置断点即可。

j的意义是对于不定长数组v的每一个元素的常引用的遍历。

且

$$j = \pi - 3 + i, i = 0, 1, 2, \ldots, 9$$

- 对于代码片段三，我们希望您能根据调试与代码中的提示修改代码让其正常运行，并告诉我们const,enum，define三者中哪一个有地址，并将其地址打印出来。

```
root@e199b59b582e:/ws/code/Assignment_1_test/question4# g++ -g test_const.cpp -o test_const
test_const.cpp: In function 'int main()':
test_const.cpp:30:17: error: lvalue required as unary '&' operand
   30 |     cout << &C::NUM1 << endl;
      |             ~~~^~~~~
```

修改后执行

```
root@e199b59b582e:/ws/code/Assignment_1_test/question4# ./test_const
hello
0x402004
3
7
10
6
```

加上gdb的调试

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /ws/code/Assignment_1_test/question4/test_const
warning: Error disabling address space randomization: Operation not permitted
warning: File "/usr/local/lib64/libstdc++.so.6.0.29-gdb.py" auto-loading has been declined by your `auto-load safe-path' set to "$debugdir:$datadir/auto-load".

Breakpoint 1, main () at test_const.cpp:24
24          cout << p << endl;
(gdb) p &p
No symbol "p" in current context.
(gdb) c
Continuing.
hello

Breakpoint 2, main () at test_const.cpp:27
27          cout << &c.NUM << endl;
(gdb) p &c.NUM
Attempt to take address of value not located in memory.
(gdb) c
Continuing.
0x402004

Breakpoint 3, main () at test_const.cpp:29
29          cout << C::NUM1 << endl;
(gdb) p &C::NUM1
Can't take address of "C::NUM1" which isn't an lvalue.
(gdb)
```

发现都不能用命令`(gdp) p`打印出来，原因如图。

发现const的是有地址的，对于#define的p，在打印p的语句下面加一句打印&p发现可行，但是#define应该是没有地址的，能打印出的那个仅仅是字符串"hello"的首地址。enum类型没有地址。