

[toc]

课上测试

ch03

作业题目：SM3算法测试

完成下面任务（14分） 1.在 Ubuntu 或 openEuler 中完成任务（推荐openEuler） 2.生成一个文档 sn.txt,内容为全班同学的 8 位学号，把你的学号排到第一个 3.使用 GmSSL 编程求 sn.txt的 Hash 值，提交代码或代码链接，以及编译运行过程（文本或截图）（10 分） 4.使用 OpenSSL 命令验证你的计算结果的正确性（4 分）

作业提交要求 (1')

0. 记录实践过程和 AI 问答过程，尽量不要截图，给出文本内容
1. (选做)推荐所有作业托管到 [gitee](#)或 [github](#) 上
2. (必做)提交作业 markdown文档，命名为“学号-姓名-作业题目.md”
3. (必做)提交作业 markdown文档转成的 PDF 文件，命名为“学号-姓名-作业题目.pdf”

作业内容

- 生成一个文档 sn.txt,内容为全班同学的 8 位学号，把你的学号排到第一个

```
root@Youer:~/TestInClass/ClassTest# mkdir testSM3
root@Youer:~/TestInClass/ClassTest# git init
hint: Using 'master' as the name for the initial branch. This default branch
name
hint: is subject to change. To configure the initial branch name to use in
all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this
command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /root/TestInClass/ClassTest/.git/
root@Youer:~/TestInClass/ClassTest# git add .
root@Youer:~/TestInClass/ClassTest# git commit -m "Init Commit"
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
root@Youer:~/TestInClass/ClassTest# tree
.
└─ testSM3
```

```
1 directory, 0 files
root@Youer:~/TestInClass/ClassTest# cd testSM3
root@Youer:~/TestInClass/ClassTest/testSM3# nano sn.txt
root@Youer:~/TestInClass/ClassTest/testSM3# git add .
root@Youer:~/TestInClass/ClassTest/testSM3# git commit -m "SM3:create
sn.txt"
[master (root-commit) fd9d683] SM3:create sn.txt
1 file changed, 29 insertions(+)
create mode 100644 testSM3/sn.txt
```

- 使用 GmSSL 编程求 sn.txt 的 Hash 值，提交代码或代码链接，以及编译运行过程

- 具体过程:

```
root@Youer:~/TestInClass/ClassTest/testSM3# nano hash_sn.c
root@Youer:~/TestInClass/ClassTest/testSM3# gcc -o hash_sn hash_sn.c -
I/usr/local/include -lgmssl
root@Youer:~/TestInClass/ClassTest/testSM3# ls
hash_sn hash_sn.c sn.txt
```

- GmSSL 编程代码:

```
#include <stdio.h>
#include <stdlib.h>
#include <gmssl/sm3.h>

int main() {
    FILE *file = fopen("sn.txt", "rb");
    if (!file) {
        perror("Unable to open file");
        return EXIT_FAILURE;
    }

    SM3_CTX sm3_ctx;
    unsigned char hash[SM3_DIGEST_SIZE]; // 使用 SM3_DIGEST_SIZE

    sm3_init(&sm3_ctx); // 注意函数名小写

    unsigned char buffer[1024];
    size_t bytesRead;
    while ((bytesRead = fread(buffer, 1, sizeof(buffer), file)) > 0) {
        sm3_update(&sm3_ctx, buffer, bytesRead); // 注意函数名小写
    }

    sm3_finish(&sm3_ctx, hash); // 注意函数名小写
    fclose(file);
```

```
printf("Hash value: ");
for (int i = 0; i < SM3_DIGEST_SIZE; i++) { // 使用 SM3_DIGEST_SIZE
    printf("%02x", hash[i]);
}
printf("\n");

return EXIT_SUCCESS;
}
```

- [hash_sn.c代码链接](#)

- 使用 OpenSSL 命令验证你的计算结果的正确性

```
root@Youer:~/TestInClass/ClassTest/testSM3# ./hash_sn
Hash value: a24b1d2b5e2e8a96f2672ad3a4d4ea72607ecd03f145040b0c7ff6e622cb82c0
root@Youer:~/TestInClass/ClassTest/testSM3# openssl dgst -sm3 sn.txt
SM3(sn.txt)= a24b1d2b5e2e8a96f2672ad3a4d4ea72607ecd03f145040b0c7ff6e622cb82c0
```

显然两者一致。我们也可以通过将输出存储到文件，再比较文件的方式进行：（但需要手动去除哈希值前的前缀）

```
root@Youer:~/TestInClass/ClassTest/testSM3# ./hash_sn > gmssl_code_hash.txt
root@Youer:~/TestInClass/ClassTest/testSM3# openssl dgst -sm3 sn.txt >
openssl_command_hash.txt
root@Youer:~/TestInClass/ClassTest/testSM3# nano gmssl_code_hash.txt
root@Youer:~/TestInClass/ClassTest/testSM3# nano openssl_command_hash.txt
root@Youer:~/TestInClass/ClassTest/testSM3# diff openssl_command_hash.txt
gmssl_code_hash.txt
root@Youer:~/TestInClass/ClassTest/testSM3#
```

- 通过AI进行学习：
 - [AI链接](#)
 - 实际对话：
 - AI的回复

如果你想使用 SM3 算法，可以通过 `sm3.h` 头文件来实现。以下是一个使用 SM3 算法计算文件哈希值的示例代码：

```
### 示例代码

```c
#include <stdio.h>
#include <stdlib.h>
#include <gmssl/sm3.h>
```

```

int main() {
 FILE *file = fopen("sn.txt", "rb");
 if (!file) {
 perror("Unable to open file");
 return EXIT_FAILURE;
 }

 SM3_CTX sm3_ctx;
 unsigned char hash[SM3_DIGEST_LENGTH];

 SM3_Init(&sm3_ctx);

 unsigned char buffer[1024];
 size_t bytesRead;
 while ((bytesRead = fread(buffer, 1, sizeof(buffer), file)) >
0) {
 SM3_Update(&sm3_ctx, buffer, bytesRead);
 }

 SM3_Final(hash, &sm3_ctx);
 fclose(file);

 printf("Hash value: ");
 for (int i = 0; i < SM3_DIGEST_LENGTH; i++) {
 printf("%02x", hash[i]);
 }
 printf("\n");

 return EXIT_SUCCESS;
}
...

```

#### ### 编译命令

确保在编译时链接 GmSSL 库，可以使用以下命令：

```

```bash
gcc -o hash_sm3 hash_sm3.c -I/usr/local/include -lgmssl
```

```

#### ### 使用方法

1. 确保你的输入文件 `sn.txt` 存在，并且可以读取。
2. 编译完成后，运行生成的可执行文件：

```

```bash
./hash_sm3
```

```

这段代码会读取 `sn.txt` 文件并计算其 SM3 哈希值，然后将结果打印出来。

希望这能帮助你顺利实现 SM3 哈希功能！如果还有其他问题，欢迎继续询问。

### ■ 我的提问:

```

你的代码有bug:
root@Youer:~/TestInClass/ClassTest/testSM3# gcc -o hash_sn
hash_sn.c -I/usr/local/include -lgmssl
hash_sn.c: In function 'main':
hash_sn.c:13:24: error: 'SM3_DIGEST_LENGTH' undeclared (first use
in this function); did you mean 'SM3_DIGEST_CTX'?
13 | unsigned char hash[SM3_DIGEST_LENGTH];
 | ^~~~~~
 | SM3_DIGEST_CTX
hash_sn.c:13:24: note: each undeclared identifier is reported only
once for each function it appears in
hash_sn.c:15:5: warning: implicit declaration of function
'SM3_Init'; did you mean 'sm3_init'? [-Wimplicit-function-declaration]
15 | SM3_Init(&sm3_ctx);
 | ^~~~~~
 | sm3_init
hash_sn.c:20:9: warning: implicit declaration of function
'SM3_Update'; did you mean 'sm3_update'? [-Wimplicit-function-
declaration]
20 | SM3_Update(&sm3_ctx, buffer, bytesRead);
 | ^~~~~~
 | sm3_update
hash_sn.c:23:5: warning: implicit declaration of function
'SM3_Final' [-Wimplicit-function-declaration]
23 | SM3_Final(hash, &sm3_ctx);
 | ^~~~~~

```

### ■ AI的回复:

看起来你遇到了与函数和常量名称不匹配的问题。这可能是由于使用的库版本或头文件不同。根据 GmSSL 的实现，函数名称可能有所不同。请尝试以下代码，这将使用正确的函数名和常量：

```

修正后的示例代码

```c
#include <stdio.h>
#include <stdlib.h>
#include <gmssl/sm3.h>

int main() {
    FILE *file = fopen("sn.txt", "rb");
    if (!file) {
        perror("Unable to open file");
        return EXIT_FAILURE;
    }

    SM3_CTX sm3_ctx;

```

```

        unsigned char hash[SM3_DIGEST_SIZE]; // 使用 SM3_DIGEST_SIZE

        sm3_init(&sm3_ctx); // 注意函数名小写

        unsigned char buffer[1024];
        size_t bytesRead;
        while ((bytesRead = fread(buffer, 1, sizeof(buffer), file)) >
0) {
            sm3_update(&sm3_ctx, buffer, bytesRead); // 注意函数名小写
        }

        sm3_finish(&sm3_ctx, hash); // 注意函数名小写
        fclose(file);

        printf("Hash value: ");
        for (int i = 0; i < SM3_DIGEST_SIZE; i++) { // 使用
SM3_DIGEST_SIZE
            printf("%02x", hash[i]);
        }
        printf("\n");

        return EXIT_SUCCESS;
    }
    ...

```

编译命令

同样，使用以下命令进行编译：

```

```bash
gcc -o hash_sn hash_sn.c -I/usr/local/include -lgmssl
```

```

说明

- 将 `SM3_DIGEST_LENGTH` 替换为 `SM3_DIGEST_SIZE`。
- 确保所有函数调用（`sm3_init`，`sm3_update`，和 `sm3_finish`）都是小写。

如果还有其他问题，请随时告诉我！

- [作业文档链接](#)