

APPLIED
CRYPTOGRAPHY
LAB3
PSUEDO RANDOM
NUMBER
GENERATION

NAME: VISHWAS M

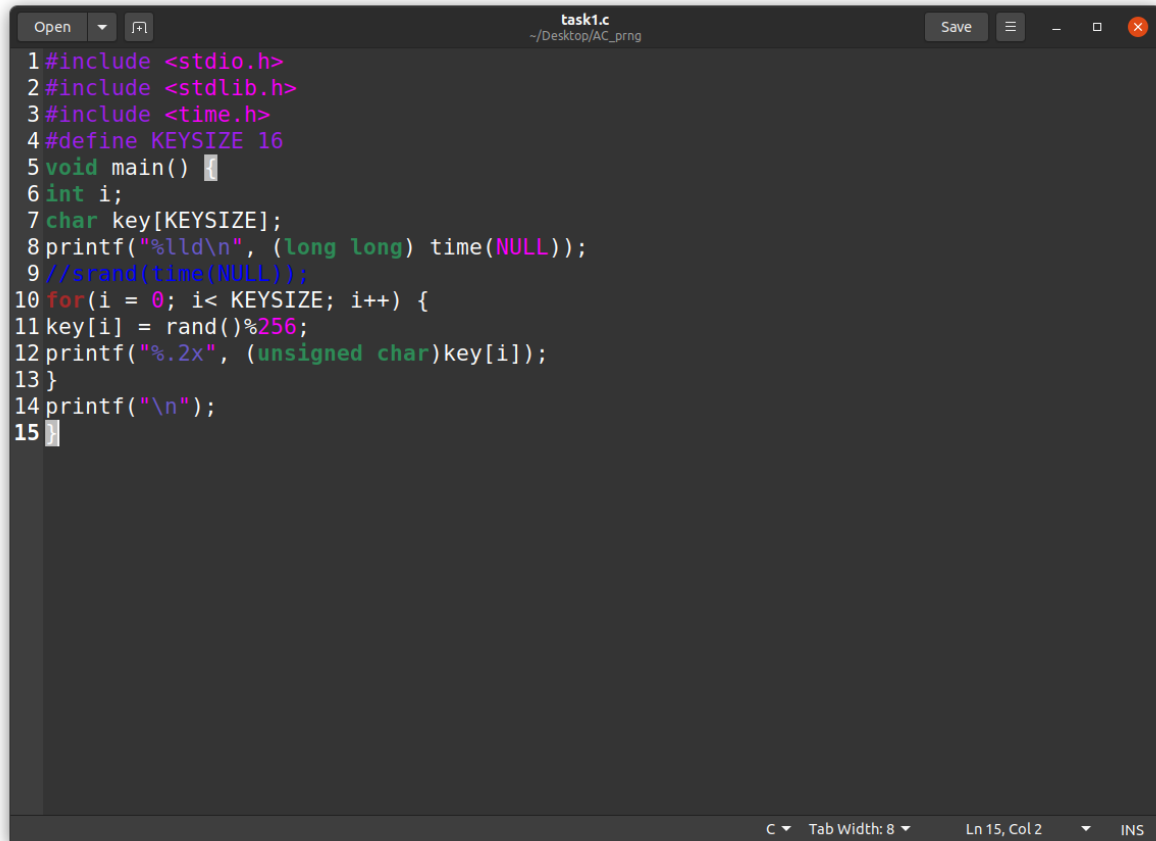
SEC: F

SRN: PES2UG20CS390

DATE: 27/09/2022

Task1:

Code:



```
1#include <stdio.h>
2#include <stdlib.h>
3#include <time.h>
4#define KEYSIZE 16
5void main() {
6    int i;
7    char key[KEYSIZE];
8    printf("%lld\n", (long long) time(NULL));
9    //srand(time(NULL));
10   for(i = 0; i< KEYSIZE; i++) {
11       key[i] = rand()%256;
12       printf("%.2x", (unsigned char)key[i]);
13   }
14   printf("\n");
15}
```

The screenshot shows a code editor window titled "task1.c" with a file path of "~/Desktop/AC_prng". The code is a C program that generates a random key of size 16. It includes headers for stdio, stdlib, and time. It defines a constant KEYSIZE as 16. The main function prints the current time, seeds the random number generator with time, and then fills a character array 'key' with random values between 0 and 255. It prints each character in hexadecimal format. The editor has a dark theme and shows line numbers from 1 to 15. The status bar at the bottom indicates "C", "Tab Width: 8", "Ln 15, Col 2", and "INS" mode.

Step1 Output:

A terminal window titled 'seed@VM: ~/.../AC_prng' showing the execution of a C program. The user runs 'gcc task1.c -o task1' and then './task1' three times. Each execution produces a long hexadecimal string. The first two strings are identical: '1664250046eca77f8c4b5dd373a74c409070f136b9'. The third string is different: '6d9cf39c8b437e06d14e3f6859087f24'.

```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM:~/.../AC_prng$ gcc task1.c -o task1
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
1664250046eca77f8c4b5dd373a74c409070f136b9
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
1664250046eca77f8c4b5dd373a74c409070f136b9
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
6d9cf39c8b437e06d14e3f6859087f24
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
16642500493fbeecca7e4c188b367c9745752b3b15c
[09/27/22] seed@VM:~/.../AC_prng$
```

As we can see here the output varies whenever we execute the above code which means that each time it is generating different key values.

Step2 Output:

After commenting:

A terminal window titled 'seed@VM: ~/.../AC_prng' showing the execution of the same C program after some code has been commented out. The user runs 'gcc task1.c -o task1' and then './task1' three times. The first two strings are identical: '166425012567c6697351ff4aec29cdbaabf2fbe346'. The third string is different: '166425012767c6697351ff4aec29cdbaabf2fbe346'.

```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM:~/.../AC_prng$ gcc task1.c -o task1
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
166425012567c6697351ff4aec29cdbaabf2fbe346
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
166425012767c6697351ff4aec29cdbaabf2fbe346
[09/27/22] seed@VM:~/.../AC_prng$ ./task1
166425012967c6697351ff4aec29cdbaabf2fbe346
[09/27/22] seed@VM:~/.../AC_prng$
```

As we can see here the output varies whenever we execute the above code which means that each time it is generating same key values.

Task2:

Code:



```
task2.c
~/Desktop/AC_prng

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 #define KEYSIZE 16
5 void main() {
6     int i, j;
7     FILE *f;
8     char key[KEYSIZE];
9     int value1, value2;
10    /* use the output of the previous step as value1 and value2 respectively */
11    value1 = 1523979529;
12    value2 = 1523986729;
13    f = fopen("keys.txt", "w");
14    for(j = value1; j <= value2; j++) {
15        srand(j);
16        for(i = 0; i < KEYSIZE; i++) {
17            key[i] = rand()%256;
18            fprintf(f, "%.2x", (unsigned char)key[i]);
19        }
20        fprintf(f, "\n");
21    }
```

Step1 Output:

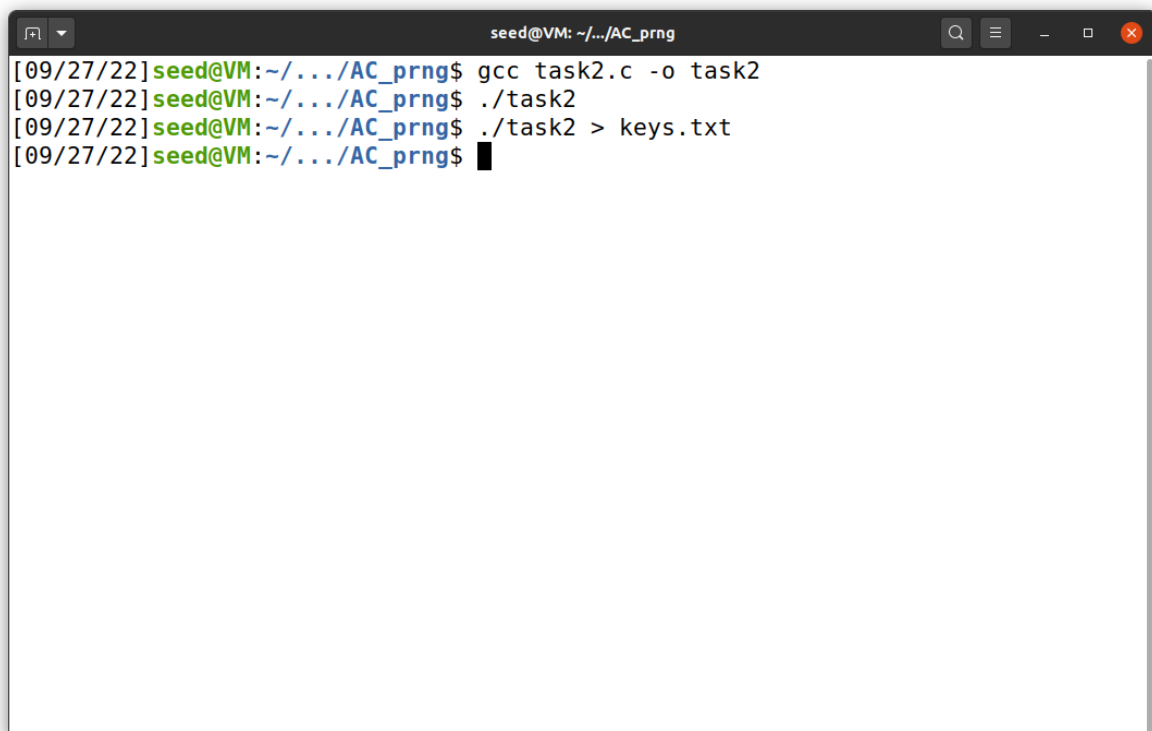
Output:



```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM:~/.../AC_prng$ date -d "2018-04-17 21:08:49" +%s
1523979529
[09/27/22] seed@VM:~/.../AC_prng$ date -d "2018-04-17 23:08:49" +%s
1523986729
[09/27/22] seed@VM:~/.../AC_prng$
```

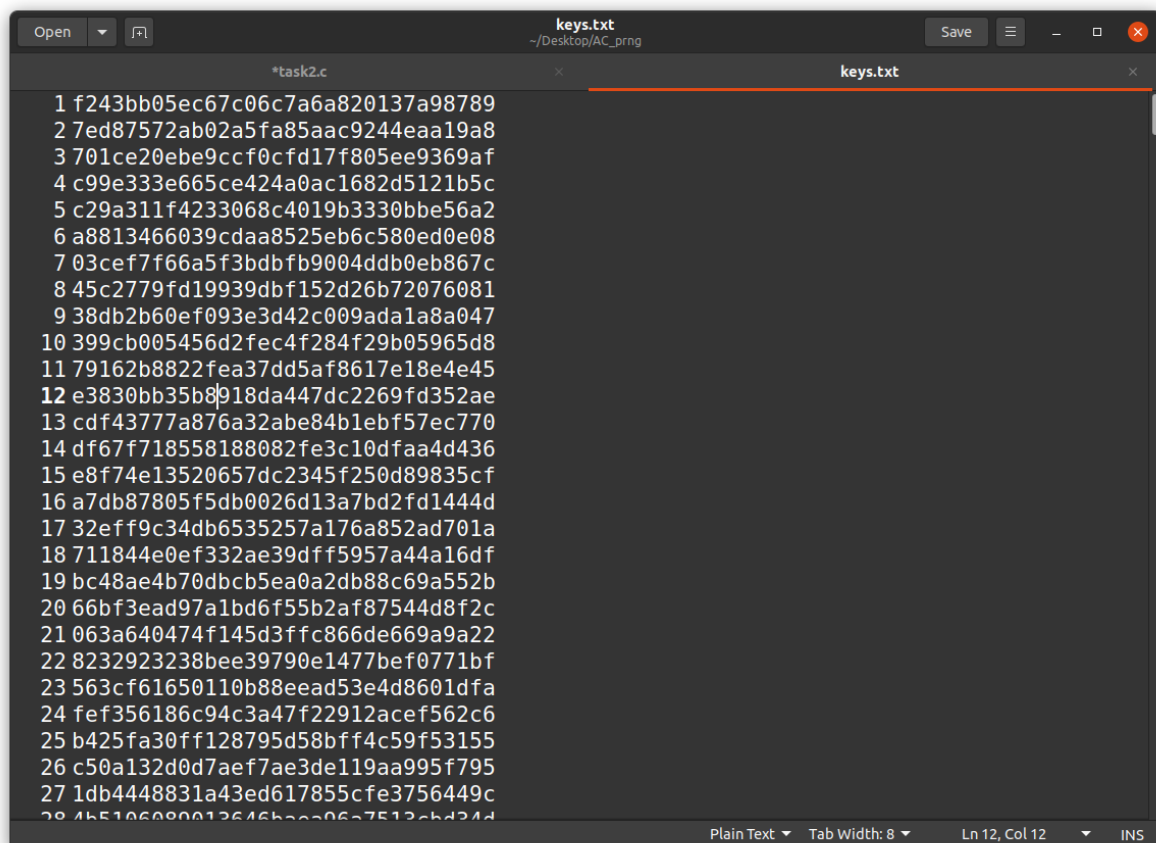
Step2 Output:

We are storing the output in a text file called keys.txt



```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM:~/.../AC_prng$ gcc task2.c -o task2
[09/27/22] seed@VM:~/.../AC_prng$ ./task2
[09/27/22] seed@VM:~/.../AC_prng$ ./task2 > keys.txt
[09/27/22] seed@VM:~/.../AC_prng$
```

Keys generated in keys.txt:



```
keys.txt
~/Desktop/AC_prng

1 f243bb05ec67c06c7a6a820137a98789
2 7ed87572ab02a5fa85aac9244eaa19a8
3 701ce20ebe9ccf0cfd17f805ee9369af
4 c99e333e665ce424a0ac1682d5121b5c
5 c29a311f4233068c4019b3330bbe56a2
6 a8813466039cdaa8525eb6c580ed0e08
7 03cef7f66a5f3bdbfb9004ddb0eb867c
8 45c2779fd19939dbf152d26b72076081
9 38db2b60ef093e3d42c009ada1a8a047
10 399cb005456d2fec4f284f29b05965d8
11 79162b8822fea37dd5af8617e18e4e45
12 e3830bb35b8918da447dc2269fd352ae
13 cdf43777a876a32abe84b1ebf57ec770
14 df67f718558188082fe3c10dfaa4d436
15 e8f74e13520657dc2345f250d89835cf
16 a7db87805f5db0026d13a7bd2fd1444d
17 32eff9c34db6535257a176a852ad701a
18 711844e0ef332ae39dff5957a44a16df
19 bc48ae4b70dbcb5ea0a2db88c69a552b
20 66bf3ead97a1bd6f55b2af87544d8f2c
21 063a640474f145d3ffc866de669a9a22
22 8232923238bee39790e1477bef0771bf
23 563cf61650110b88eead53e4d8601dfa
24 fef356186c94c3a47f22912acef562c6
25 b425fa30ff128795d58bff4c59f53155
26 c50a132d0d7aef7ae3de119aa995f795
27 1db4448831a43ed617855cfe3756449c
28 4b510600012646b000607512ebd34d
```

Step3 Output:



```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM:~/.../AC_prng$ gcc task2.c -o task2
[09/27/22] seed@VM:~/.../AC_prng$ ./task2 > keys.txt
[09/27/22] seed@VM:~/.../AC_prng$ python3 decrypt.py

Match found
Match found
key: 95fa2030e73ed3f8da761b4eb805dfd7
Ciphertext: d06bf9d0dab8e8ef880660d2af65aa82
Encrypted: d06bf9d0dab8e8ef880660d2af65aa82

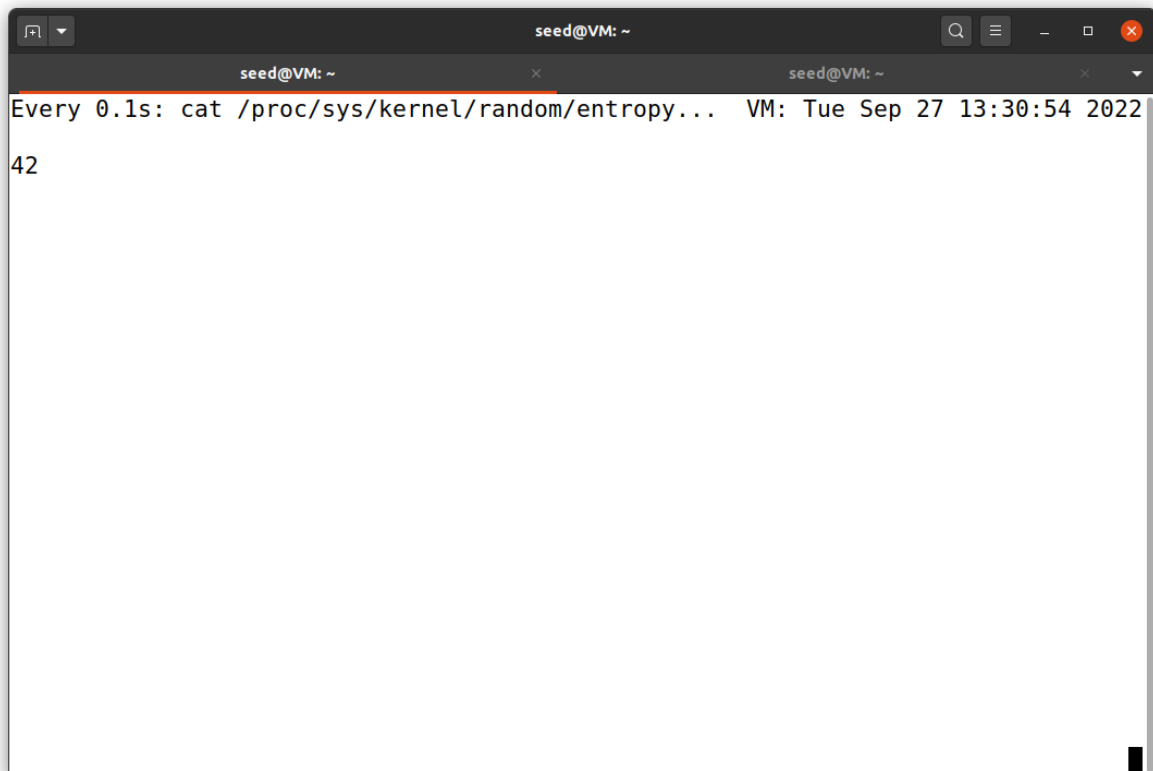
[09/27/22] seed@VM:~/.../AC_prng$
```

Task3:

A terminal window titled 'seed@VM: ~' with standard window controls. It displays a cron job output: 'Every 0.1s: cat /proc/sys/kernel/random/entropy...' followed by the date 'VM: Tue Sep 27 13:26:24 2022' and the value '2936'.

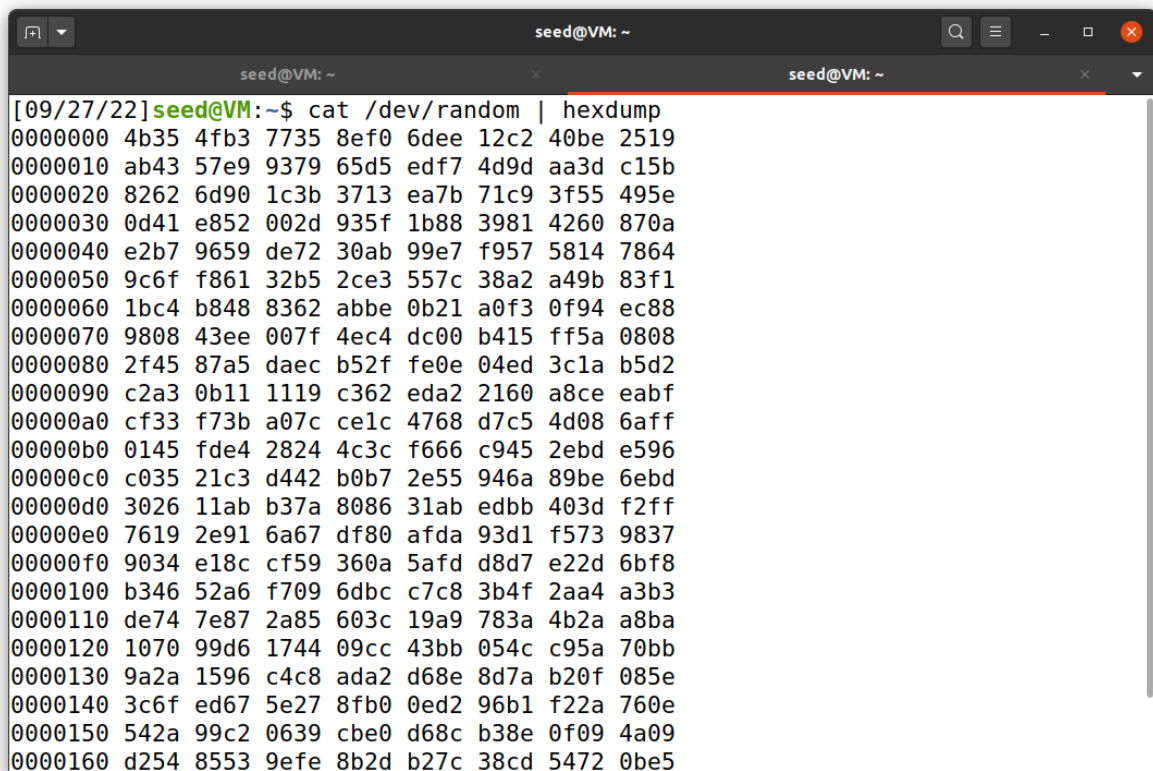
```
seed@VM: ~  
Every 0.1s: cat /proc/sys/kernel/random/entropy... VM: Tue Sep 27 13:26:24 2022  
2936
```

Task4:



A terminal window titled 'seed@VM: ~' with two tabs. The first tab shows the command 'Every 0.1s: cat /proc/sys/kernel/random/entropy...' and the output '42'. The second tab is empty. The window title bar includes standard Linux window controls and a search icon.

```
seed@VM: ~  
Every 0.1s: cat /proc/sys/kernel/random/entropy... VM: Tue Sep 27 13:30:54 2022  
42
```

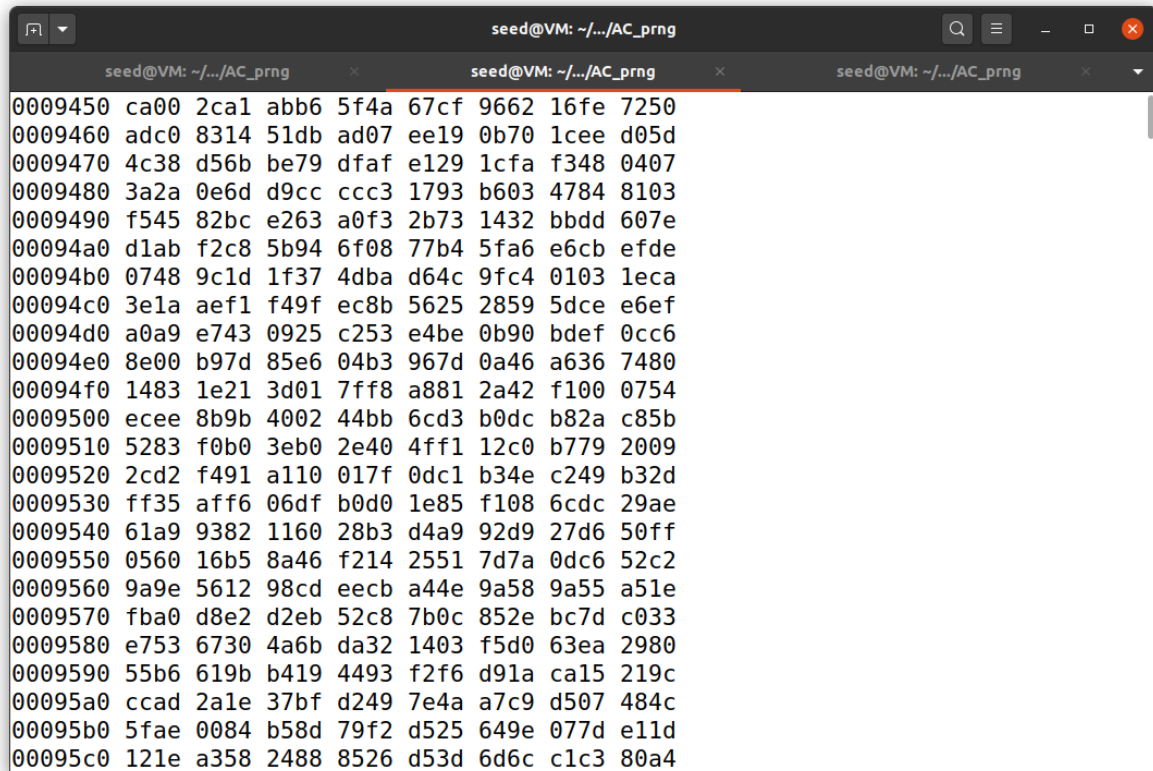


A terminal window titled 'seed@VM: ~' with two tabs. The first tab shows the command '[09/27/22]seed@VM:~\$ cat /dev/random | hexdump' and its output, which is a hexdump of random data. The second tab is empty. The window title bar includes standard Linux window controls and a search icon.

```
[09/27/22]seed@VM:~$ cat /dev/random | hexdump  
00000000 4b35 4fb3 7735 8ef0 6dee 12c2 40be 2519  
00000010 ab43 57e9 9379 65d5 edf7 4d9d aa3d c15b  
00000020 8262 6d90 1c3b 3713 ea7b 71c9 3f55 495e  
00000030 0d41 e852 002d 935f 1b88 3981 4260 870a  
00000040 e2b7 9659 de72 30ab 99e7 f957 5814 7864  
00000050 9c6f f861 32b5 2ce3 557c 38a2 a49b 83f1  
00000060 1bc4 b848 8362 abbe 0b21 a0f3 0f94 ec88  
00000070 9808 43ee 007f 4ec4 dc00 b415 ff5a 0808  
00000080 2f45 87a5 daec b52f fe0e 04ed 3c1a b5d2  
00000090 c2a3 0b11 1119 c362 eda2 2160 a8ce eabf  
000000a0 cf33 f73b a07c ce1c 4768 d7c5 4d08 6aff  
000000b0 0145 fde4 2824 4c3c f666 c945 2ebd e596  
000000c0 c035 21c3 d442 b0b7 2e55 946a 89be 6ebd  
000000d0 3026 11ab b37a 8086 31ab edbb 403d f2ff  
000000e0 7619 2e91 6a67 df80 afda 93d1 f573 9837  
000000f0 9034 e18c cf59 360a 5afd d8d7 e22d 6bf8  
00001000 b346 52a6 f709 6dbc c7c8 3b4f 2aa4 a3b3  
00001100 de74 7e87 2a85 603c 19a9 783a 4b2a a8ba  
00001200 1070 99d6 1744 09cc 43bb 054c c95a 70bb  
00001300 9a2a 1596 c4c8 ada2 d68e 8d7a b20f 085e  
00001400 3c6f ed67 5e27 8fb0 0ed2 96b1 f22a 760e  
00001500 542a 99c2 0639 cbe0 d68c b38e 0f09 4a09  
00001600 d254 8553 9efe 8b2d b27c 38cd 5472 0be5
```

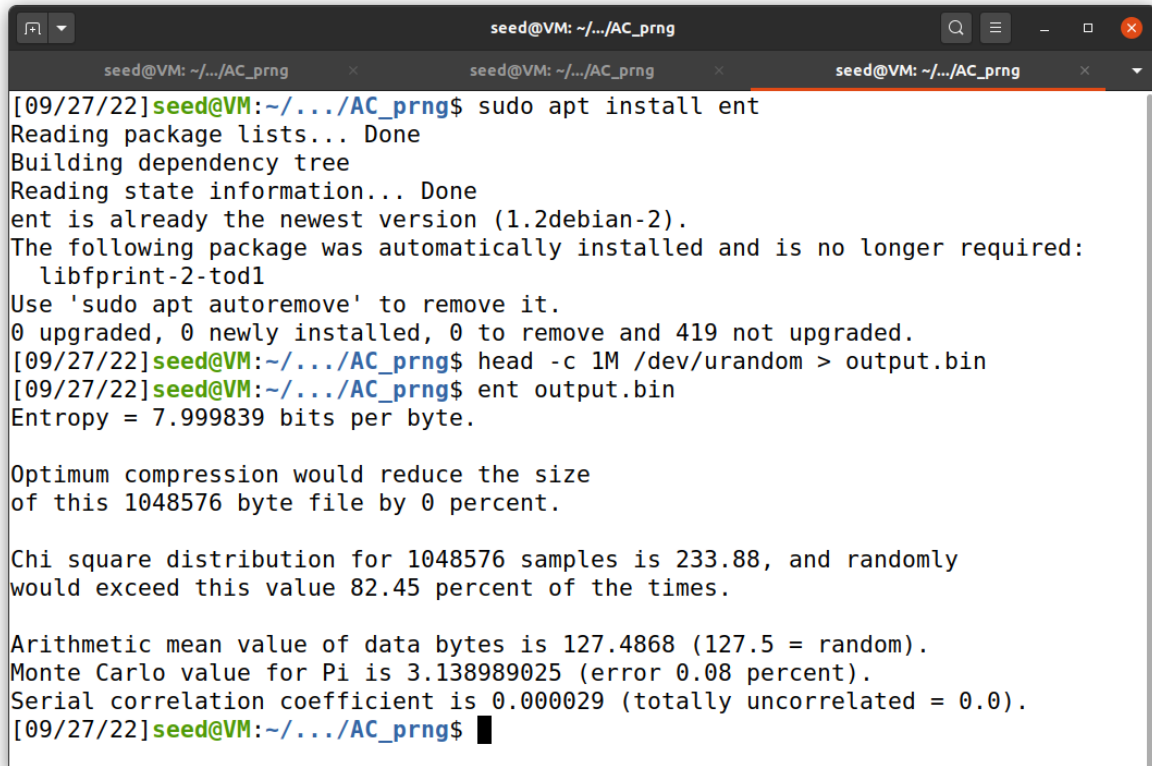

Task5:

Step1:



```
seed@VM: ~/.../AC_prng
0009450 ca00 2ca1 abb6 5f4a 67cf 9662 16fe 7250
0009460 adc0 8314 51db ad07 ee19 0b70 1cee d05d
0009470 4c38 d56b be79 dfaf e129 1cfa f348 0407
0009480 3a2a 0e6d d9cc ccc3 1793 b603 4784 8103
0009490 f545 82bc e263 a0f3 2b73 1432 bbdd 607e
00094a0 d1ab f2c8 5b94 6f08 77b4 5fa6 e6cb efde
00094b0 0748 9c1d 1f37 4dba d64c 9fc4 0103 1eca
00094c0 3e1a aef1 f49f ec8b 5625 2859 5dce e6ef
00094d0 a0a9 e743 0925 c253 e4be 0b90 bdef 0cc6
00094e0 8e00 b97d 85e6 04b3 967d 0a46 a636 7480
00094f0 1483 1e21 3d01 7ff8 a881 2a42 f100 0754
0009500 ecee 8b9b 4002 44bb 6cd3 b0dc b82a c85b
0009510 5283 f0b0 3eb0 2e40 4ff1 12c0 b779 2009
0009520 2cd2 f491 a110 017f 0dc1 b34e c249 b32d
0009530 ff35 aff6 06df b0d0 1e85 f108 6cdc 29ae
0009540 61a9 9382 1160 28b3 d4a9 92d9 27d6 50ff
0009550 0560 16b5 8a46 f214 2551 7d7a 0dc6 52c2
0009560 9a9e 5612 98cd eecb a44e 9a58 9a55 a51e
0009570 fba0 d8e2 d2eb 52c8 7b0c 852e bc7d c033
0009580 e753 6730 4a6b da32 1403 f5d0 63ea 2980
0009590 55b6 619b b419 4493 f2f6 d91a ca15 219c
00095a0 ccad 2a1e 37bf d249 7e4a a7c9 d507 484c
00095b0 5fae 0084 b58d 79f2 d525 649e 077d e11d
00095c0 121e a358 2488 8526 d53d 6d6c c1c3 80a4
```

Step2:



```
seed@VM: ~/.../AC_prng
[09/27/22]seed@VM:~/.../AC_prng$ sudo apt install ent
Reading package lists... Done
Building dependency tree
Reading state information... Done
ent is already the newest version (1.2debian-2).
The following package was automatically installed and is no longer required:
  libfprint-2-tod1
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 419 not upgraded.
[09/27/22]seed@VM:~/.../AC_prng$ head -c 1M /dev/urandom > output.bin
[09/27/22]seed@VM:~/.../AC_prng$ ent output.bin
Entropy = 7.999839 bits per byte.

Optimum compression would reduce the size
of this 1048576 byte file by 0 percent.

Chi square distribution for 1048576 samples is 233.88, and randomly
would exceed this value 82.45 percent of the times.

Arithmetic mean value of data bytes is 127.4868 (127.5 = random).
Monte Carlo value for Pi is 3.138989025 (error 0.08 percent).
Serial correlation coefficient is 0.000029 (totally uncorrelated = 0.0).
[09/27/22]seed@VM:~/.../AC_prng$
```

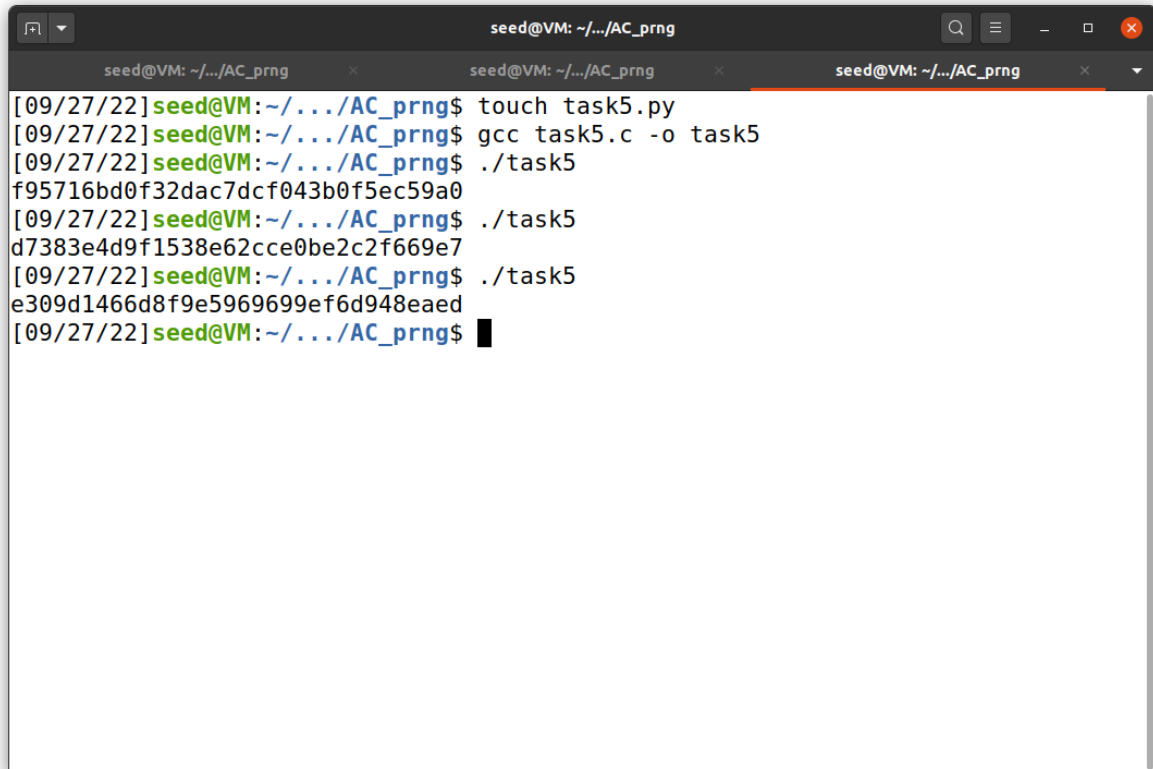
Step3:

Code:



```
task5.c
~/Desktop/AC_prng
1#include <stdio.h>
2#include <stdlib.h>
3#include <time.h>
4#define KEYSIZE 16
5void main()
6int i;
7FILE *random;
8unsigned char*key = (unsigned char*) malloc(sizeof(unsigned char) * KEYSIZE);
9random = fopen("/dev/urandom", "r");
10for(i = 0; i< KEYSIZE; i++) {
11fread(key, sizeof(unsigned char) * KEYSIZE, 1, random);
12printf("%.2x", *key);
13}printf("\n");
14fclose(random);
15
```

Output:



```
seed@VM: ~/.../AC_prng
[09/27/22] seed@VM: ~/.../AC_prng$ touch task5.py
[09/27/22] seed@VM: ~/.../AC_prng$ gcc task5.c -o task5
[09/27/22] seed@VM: ~/.../AC_prng$ ./task5
f95716bd0f32dac7dcf043b0f5ec59a0
[09/27/22] seed@VM: ~/.../AC_prng$ ./task5
d7383e4d9f1538e62cce0be2c2f669e7
[09/27/22] seed@VM: ~/.../AC_prng$ ./task5
e309d1466d8f9e5969699ef6d948eaed
[09/27/22] seed@VM: ~/.../AC_prng$
```