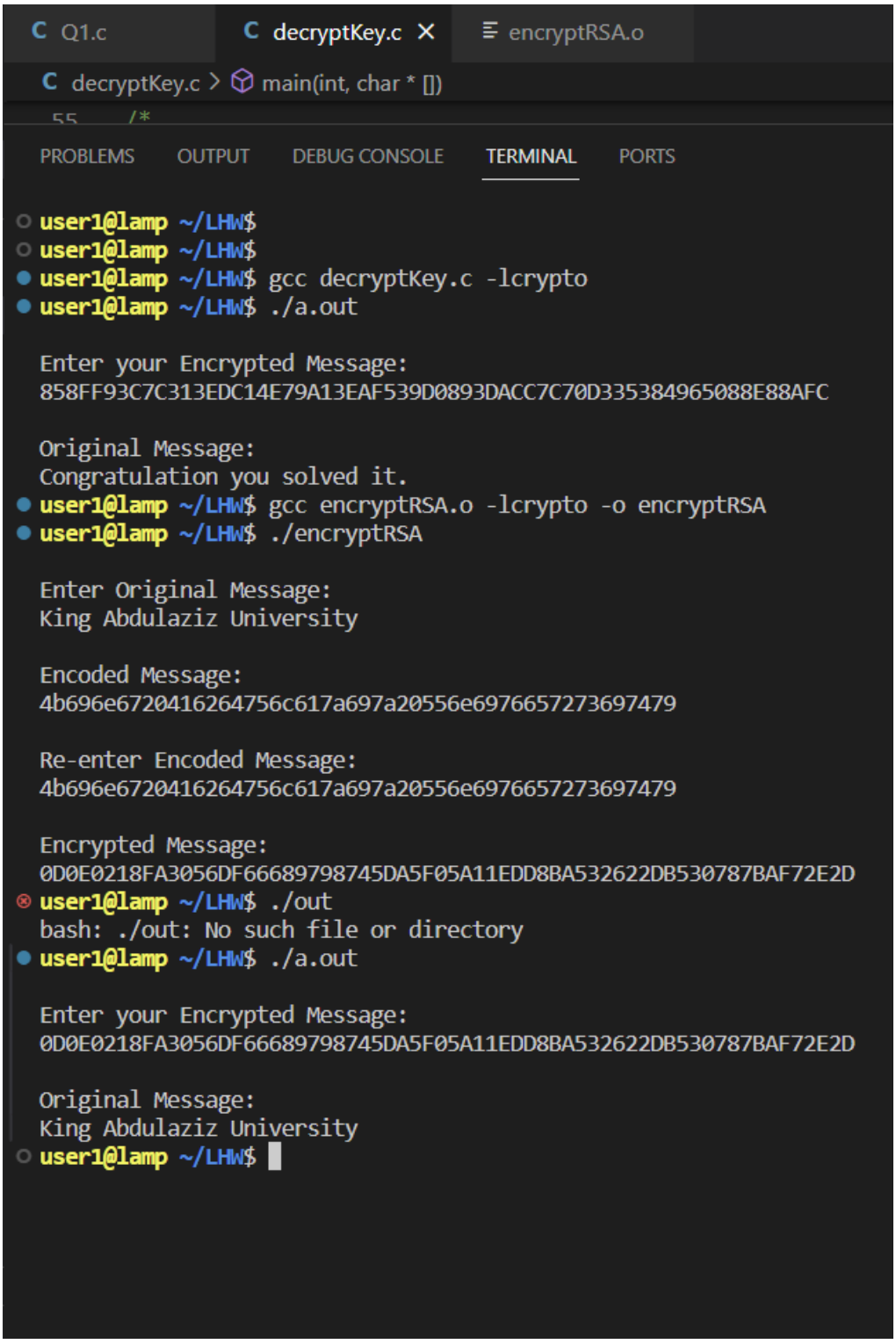
OPERATING SYSTEM

[LAB Homework Q1]

Talal Mohammed Alqarni

[2036183]

Output:


Q1.c
decryptKey.c
encryptRSA.o
decryptKey.c > main(int, char * [])
55 /*
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
user1@lamp ~/LHW$
user1@lamp ~/LHW$
user1@lamp ~/LHW$ gcc decryptKey.c -lcrypto
user1@lamp ~/LHW$ ./a.out
Enter your Encrypted Message:
858FF93C7C313EDC14E79A13EAF539D0893DACC7C70D335384965088E88AFC
Original Message:
Congratulation you solved it.
user1@lamp ~/LHW$ gcc encryptRSA.o -lcrypto -o encryptRSA
user1@lamp ~/LHW$ ./encryptRSA
Enter Original Message:
King Abdulaziz University
Encoded Message:
4b696e6720416264756c617a697a20556e6976657273697479
Re-enter Encoded Message:
4b696e6720416264756c617a697a20556e6976657273697479
Encrypted Message:
0D0E0218FA3056DF66689798745DA5F05A11EDD8BA532622DB530787BAF72E2D
user1@lamp ~/LHW$ ./out
bash: ./out: No such file or directory
user1@lamp ~/LHW$ ./a.out
Enter your Encrypted Message:
0D0E0218FA3056DF66689798745DA5F05A11EDD8BA532622DB530787BAF72E2D
Original Message:
King Abdulaziz University
user1@lamp ~/LHW$

Code:

```c
/*
*
*
* Description:
* RSA DecryptionKey using OpenSSL library and Python encode/decode
* */
#include <stdio.h>
#include <string.h>
#include <openssl/bn.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>

void printBN(char *msg, BIGNUM *tmp){
char *number_str = BN_bn2hex(tmp); // Convert BIGNUM to hex
printf("%s%s\n", msg, number_str); // Print hex
OPENSSL_free(number_str); // Free memory
}
int main(int argc, char *argv[]){
BN_CTX *ctx = BN_CTX_new();

/*
* Here initialize all needed BIGNUM variables
* 1- Encryption Key variable
* 2- Decryption Key variable
* 3- product of large prime numbers p and q
* 4- Totient of (n) Euler's totient function
* 5- Encrypted Message variable
* 6- Decrypted Ciphertext variable
* */

BIGNUM *EK_variable = BN_new();
BIGNUM *DK_variable = BN_new();
BIGNUM *Product_of_pq = BN_new();
BIGNUM *Phin = BN_new();
BIGNUM *EM = BN_new();
BIGNUM *DC = BN_new();

// Find DecryptionKey Key (d) using (e) and (Phin):
// 1- Assign value to (e) EncryptionKey Key from hex
BN_hex2bn(&EK_variable, "010001");
// 2- Assign value to (Phin) EncryptionKey Key from hex
BN_hex2bn(&Phin, "E103ABD94892E3E74AFD724BF28E78348D52298BD687C44DEB3A81065A7981A4");
// 3- Calculate the DecryptionKey Key (Private Key) d=e mod(Phi(n))
BN_mod_inverse(DK_variable, EK_variable, Phin, ctx);

char *CC= malloc(100 * sizeof(char));
printf("\nEnter your Encrypted Message:\n");
```

```
// Read the Encrypted Message from the user to variable CC
fgets(CC, 500, stdin);
// Assign the input value in variable (CC) to Encrypted Message variable
BN_hex2bn(&EM, CC);


/*
Decrypt ciphertext using D=C^d(mod(n)) ,
where: (D) is the Decrypted Ciphertext and (C) is the Ciphertext
*/
// Assign value to (n) product of two large prime numbers from hex
BN_hex2bn(&Product_of_pq,
"E103ABD94892E3E74AFD724BF28E78366D9676BCCC70118BD0AA1968DBB143D1");
// decrypt Ciphertext using the Private Key
BN_mod_exp(DC, EM, DK_variable, Product_of_pq, ctx);

// Convert Hex string to ASCII letters
printf("\nOriginal Message:\n");
char str1[500]="print(\"";
char *str2 = BN_bn2hex(DC);
char str3[]="\".decode(\"hex\"))";
strcat(str1,str2);
strcat(str1,str3);
char* args[]={"python2", "-c",str1, NULL};
execvp("python2", args);
return EXIT_SUCCESS;
}
```

Discussion:

All I had to do in this assignment after installing the (openssl) library is to initialize all needed BIGNUM variables then assign the necessary values. Moreover, we used the variable CC to read the encrypted message. In addition, I assigned the value of the product of the two numbers. Furthermore, I decrypted the ciphertext using the private key. Finally, I tested the code by encrypting and decrypting a message.