Lab Assignment 1

Osama Al Zahrani 2044104 EE-463

Q1)

C Code

```
#include <stdio.h>
#include <string.h>

#define ALPHABET_SIZE 26

void print_password(char password[]) {
    printf("%s\n", password);
}

void generate_passwords(char password[], int index, int used[]) {
    if (index == 4) {
        print_password(password);
        return;
    }

    for (int i = 0; i < ALPHABET_SIZE; i++) {
        if (!used[i]) {
            password[index] = 'a' + i;
            used[i] = 1;
            generate_passwords(password, index + 1, used);
            used[i] = 0;
        }
    }
}

int main() {
    char password[5];
    int used[ALPHABET_SIZE] = {0};

    generate_passwords(password, 0, used);

    return 0;
}
```

Bash Code

```
echo $((26*25*24*23))
```

C Code Output

```
zyxc
zyxd
zyxe
zyxf
zyxg
zyxh
zyxi
zyxj
zyxk
zyxl
zyxm
zyxn
zyxo
zyxp
zyxq
zyxr
zyxs
zyxt
zyxu
zyxv
zyxw

Osama@DESKTOP-J7N4P3D MINGW64 ~/Documents/github/first_repo/Lab_Assigment_1 (mai
$ ./Q1.exe
```

Bash Code Output

```
Osama@DESKTOP-J7N4P3D MINGW64 ~/Documents/github/first_repo/Lab_Assigment_1 (ma
n)
$ echo $((26*25*24*23))
358800
```

Q2)

C Code

```
/*
* Author:
* Description:
* RSA Decryption 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();

BIGNUM *e = BN_new(); // Encryption Key
BIGNUM *d = BN_new(); // Decryption Key
BIGNUM *n = BN_new(); // product of large prime numbers p and q
BIGNUM *phin = BN_new(); // Totient of (n) Euler's totient function
BIGNUM *C = BN_new(); // Encrypted Message
BIGNUM *D = BN_new(); // Decrypted Ciphertext
// Find Decryption Key (d) using (e) and (Phin):
// 1- Assign value to (e) Encryption Key from hex
BN_hex2bn(&e, "010001");
```

```
// 2- Assign value to (Phin) Encryption Key from hex
BN_hex2bn(&phin,
"E103ABD94892E3E74AFD724BF28E78348D52298BD687C44DEB3A81065A7981A4");
// 3- Calculate the Decryption Key (Private Key) d=e mod(Phi(n))
BN_mod_inverse(d, e, phin, ctx); // d = e^-1 mod phin
char *CC= malloc(100 * sizeof(char));
printf("\nEnter your Encrypted Message:\n");
// Read the Encrypted Message from the user to variable CC
scanf("%s", CC);
// Assign the input value in variable (CC) to Encrypted Message variable
BN_hex2bn(&C, 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(&n,
"E103ABD94892E3E74AFD724BF28E78366D9676BCCC70118BD0AA1968DBB143D1");
// decrypt Ciphertext using the Private Key
BN_mod_exp(D, C, d, n, ctx); // D = C^d mod n
// Convert Hex string to ASCII letters
printf("\nOriginal Message:\n");
char str1[500]="print(\"";
char *str2 = BN_bn2hex(D);
char str3[]="\".decode(\"hex\"))";
strcat(str1,str2);
strcat(str1,str3);
char* args[]={"python2", "-c",str1, NULL};
execvp("python2", args);
return EXIT_SUCCESS;
}
```

C Code Output