Name: Ekansh Kumar
Reg. No: 23BCE0414

Q-1)

RSA Algorithm -

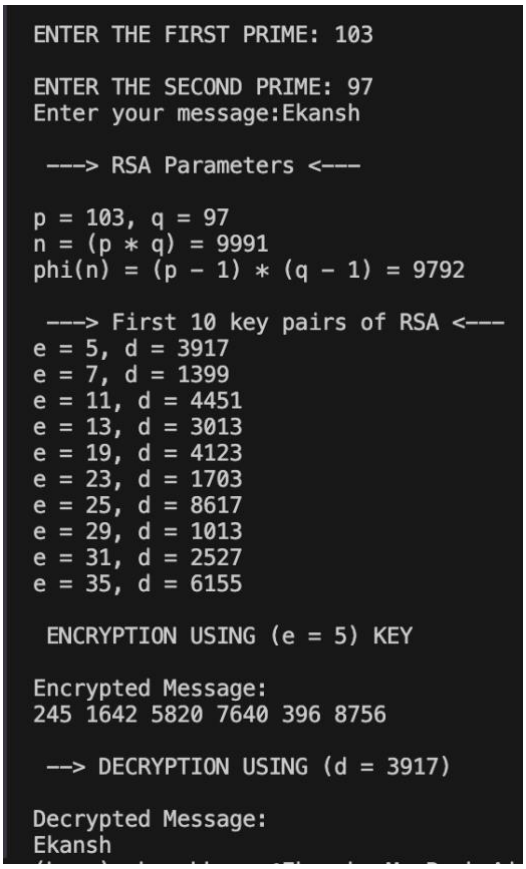
```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

long int p, q, n, t, i, en[100], e[100], d[100], m[100],temp[100];
char msg[100];

int prime(long int); long int
gcd(long int, long int); void
find_keys(); int
compute_d(long int); void
encrypt(); void decrypt();


int main() { printf("\nENTER THE FIRST PRIME:
    "); scanf("%ld", &p);

    if (!prime(p)) {
        printf("NOT A PRIME NUMBER\n");
        exit(1); }

    printf("\nENTER THE SECOND PRIME: "); scanf("%ld",
    &q);

    if (!prime(q) || p == q) {
        printf("NOT A DISTINCT / VALID PRIME NUMBER\n");
    exit(1); }

    printf("Enter your message:");
    scanf("  %[^\n]s", msg);

    for (int i=0; msg[i] != '\0'; i++) { m[i]
        = msg[i];
    }


    printf("\n ---> RSA Parameters <---\n\n");
    n = p * q; t = (p - 1) * (q - 1); printf("p =
```

```
    %ld, q = %ld\n", p, q); printf("n = (p * q) =
    %ld\n", n); printf("phi(n) = (p - 1) * (q - 1)
    = %ld\n", t); find_keys();

    printf("\n ---> First 10 key pairs of RSA <---\n");
    for  (int i = 0; i < 10 && e[i] != 0; i++) { printf("e =
    %ld, d = %ld\n", e[i], d[i]);
    }

    encrypt();
    decrypt();
    return 0;
}

void encrypt() { long int
    pt, key = e[0];
    printf("\n ENCRYPTION USING (e = %ld) KEY\n", key); int len
    = strlen(msg);

    for (int i = 0; i < len; i++) {
        pt = m[i]; long int
        result = 1;

        for (long int j = 0; j < key; j++) { result
            = (result * pt)  % n;
        }

        temp[i] = result;
        en[i] = result;

    }
    en[len] = -1; printf("\nEncrypted
    Message: \n"); for (int i = 0; en[i]
    != -1; i++) { printf("%ld ", en[i]);
    }
    printf("\n");
}

void decrypt() { long int
    ct, key = d[0]; int i = 0;

    printf("\n --> DECRYPTION USING (d = %ld)\n", key);

    while(en[i] != -1) { ct
        = temp[i];
```

```
            long int result = 1; for (long int j
            = 0; j < key; j++) { result = (result
            * ct) % n;
            }
            m[i] = result;
      i++; }
      m[i] = -1;

      printf("\nDecrypted Message: \n");
      for (i = 0; m[i] != -1; i++) {
            printf("%c", m[i]);
      }
      printf("\n");

}

void find_keys() { int
      k = 0;
      for (long int i = 2; i < t && k < 99; i++) { if (i
            == p || i == q) {
                  continue;
            }
            if (gcd(i, t) == 1) {

                  e[k] = i; d[k] =
                  compute_d(e[k]);
                  if (d[k] > 0) { k++;
                  }
            }
      }
}

int compute_d(long int e_val) { long
      int k; for (k = 1; k < 100000; k++)
      { if ((k * t + 1) % e_val == 0) {
      return (k * t  + 1) / e_val;
            }
      }
      return -1;
}

int prime(long int a) { if (a <
      2) return 0; if (a == 2)
      return 1; if (a % 2 == 0)
      return 0;
```

```
    for (long int k = 3; k < sqrt(a); k += 2) {
        if (a % k  == 0){
            return 0;
        }
    }
    return 1; }

long int gcd(long int a, long int b) {
    while (b != 0) { long
        int temp = b; b =
        a % b; a = temp;
    }
    return a;
}
```

Output -



Q-2) Diffi Hellman Key Exchange -

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int prime(long int); long int modEx(long
int, long int, long int); int main() {

    int xa, xb, g, p; printf("Enter
    xa, xb, g, p\n"); scanf("%d",
    &xa); scanf("%d", &xb);
    scanf("%d", &g); scanf("%d",
    &p);
    if (!prime(xa) || !prime(p) || !prime(g) ||
    !prime(xb)){ printf("Invalid Inputs\n"); exit(1); }
    long int ya = modEx(g, xa, p);
    long int yb = modEx(g, xb, p);
    long sa = modEx(yb, xa, p); long
    sb = modEx(ya, xb, p);
    printf("ya = %ld, yb = %ld, sa = %ld, sb = %ld\n", ya, yb, sa, sb); if
    (sa == sb) { printf("Exchange successful!\n");
    }
    else { printf("Exchange
        Unsuccessful!\n");
    } return
0; }

long int modEx(long int a, long e, long n)  {
    long int result = 1; for (long int j = 0; j <
    e; j++) { result = (result * a)  % n;
        }
    return result;
}


int prime(long int a) { if (a <
    2) return 0; if (a == 2)
    return 1; if (a % 2 == 0)
    return 0;
      for (long int k = 3; k < sqrt(a); k += 2) {
        if (a % k  == 0){
            return 0;
        }
    }
```

```
    return 1;
}
```

Output -