Universidade Federal do Espírito Santo – Centro Tecnológico
Departamento de Informática
Prof. Vinícius Fernandes Soares Mota

UFES

https://pt.wikipedia.org/wiki/Troca_de_chaves_de_Diffie%E2%80%93Hellman

Alice e Bob queriam trocar mensagens codificadas sem que fosse necessário que um soubesse a senha (chave) do outro. Alice então sugeriu para Bob o uso do algoritmo Diffie-Hellman, inventado em 1976. Alice então explicou que este foi um dos primeiros algoritmos para troca de chaves, que permite que dois ou mais usuários tenham uma chave secreta em comum, sem conhecer as chaves privadas. Este é o princípio do conceito de chave pública e privada, no qual usuários que querem trocar mensagens compartilham apenas sua chave pública.
O algoritmo funciona da seguinte forma, primeiramente, define-se um número primo (p)e uma base (g) em comum, cada um define sua chave privada (a) a partir destes, calcula-se as chaves públicas, como CP = $g^a \bmod p$

O exemplo abaixo mostra como funcionaria a troca de chaves entre Bob e Alice, onde os valores públicos estão em azul e os valores secretos estão em **vermelho e destacados**:

1. Alice e Bob entram em acordo para usar um número primo p=23 e como base g=5.
2. Alice escolhe um inteiro secreto ***a*=6**, e então calcula sua chave pública e envia a Bob $A = g^a \bmod p$
   - $A = 5^6 \bmod 23$
   - $A = \mathbf{15.625} \bmod 23$
   - $A = 8$
3. Bob escolhe um inteiro secreto ***b*=15**, e então calcula sua chave pública e envia a Alice $B = g^b \bmod p$
   - $B = 5^{15} \bmod 23$
   - $B = \mathbf{30.517.578.125} \bmod 23$
   - $B = 19$
4. Alice calcula **s** = $B^a \bmod p$
   - **s** = $19^6 \bmod 23$
   - **s** = $\mathbf{47.045.881} \bmod 23$
   - **s** = **2**

Com a chave B de Bob, Alice calculou s e pode usar s para criptografar/descriptografar a mensagem.

5. Bob calcula **s** = $A^b \bmod p$
   - **s** = $8^{15} \bmod 23$
   - **s** = $\mathbf{35.184.372.088.832} \bmod 23$
   - **s** = **2**

Com a chave A de Alice, Bob calculou s e pode usar *s* para criptografar a mensagem.

A mensagem será criptografada “somando” as letras minúsculas do texto a essa chave s para que o texto fique codificado. As letras maiúsculas são codificadas com a chave 2 vezes maior que a chave normal.

Exemplo: se *s* = 2, a letra a vira c, a letra n vira p, a letra z vira b, e a letra A vira E. O programa deve decodificar o texto revertendo o texto para o original, ou seja, se s = 2 a letra c vira a, a letra b vira z, e assim sucessivamente.

Para isso, faça um programa que contenha três opções:

1 para gerar a chave pública. Recebe como entrada uma chave privada, a base e um número primo. Imprime a chave pública;
2 Codificar a mensagem. Recebe como entrada a chave privada, a base e um número primo e a chave pública com a qual será calculado a chave secreta. Ler conjunto de caracteres até encontrar um ".". Codificar e imprimir de acordo com a regra acima. Apenas caracteres do alfabeto devem ser codificados.
3 Decodificar a mensagem. Recebe como entrada a chave privada, a base e um número primo a chave pública do remetente, com a qual será calculado a chave secreta. Ler conjunto de caracteres até encontrar um ".". Decodificar e imprimir de acordo com a regra acima. Apenas caracteres do alfabeto devem ser decodificados.

Se for dada uma opção diferente das opções acima o programa deverá imprimir: "*Operacao invalida*." Se o argumento primo não for primo deverá imprimir "P precisa ser primo".

Seu código deverá testar os limites e as regras de cada entrada, por exemplo, se a entrada p é um número primo realmente. Além disso, deve ser modularizado, isto é, **deve conter,** no mínimo, o seguinte protótipo de funções:

(20%) **char** Codifica**(char** letra**, int** n**)**

Função que recebe uma letra e a chave de codificação (inteiro) e retorna a letra codificada.

(20%) **char** Decodifica **(char** letra**, int** n**)**

Função que decodifique uma letra, recebendo como parâmetro a letra e um inteiro que represente a chave de codificação e retorna a letra decodificada.

(20%) **int** CalculaChave(**int** base, **int** chave, **int** primo)

Função que recebe uma base, uma chave privada e um número primo e retorna a chave de acordo com o algoritmo Diffie-Hellman explicado acima.

(15%) **int** EhPrimo **(int** p **)**

Recebe um inteiro e retorna 1 se for primo e 0 se não for.

(10%) **long long int** exponenciacao**(int** x**, int** y**)**
retorna xy (implementar a função sem usar pow() da math.h)

(5%) **int** EhLetra **(char** c**)**

Recebe uma letra e retorna um valor que dirá se é letra ou não.

(5%) **int** EhLetraMaiuscula **(char** c**)**

Recebe uma letra e retorna 1 se ela for mauiscula e 0 se não for.

(5%) **int** EhLetraMinuscula **(char** c**)**

Recebe uma letra e retorna 1 se for minúscula e 0 se for maíuscula.

Obs.: Números e outros caracteres que por ventura apareçam no texto não deverão ser codificados.
Obs2: Não deve ser usado a função pow do math.h
Dica: O tipo long long int permite o uso de números inteiros grandes. Para imprimir um long long int use o formato "%lld".

Entrada: dois números inteiros representando modo codificador ou decodificador e a chave de codificação.
Saída: o texto codificado ou decodificado.

Exemplo de Entradas:

Entrada	**Saída esperada**
1 6 5 23	8
1 15 5 23	19
2 6 5 23 19 Ja esta tarde demais, que sono!!! zZ.	Nc guvc vctfg fgocku, swg uqpq!!! bD.
3 15 5 23 8 Nc guvc vctfg fgocku, swg uqpq!!! bD.	Ja esta tarde demais, que sono!!! zZ.
2 15 5 23 8 vai dormir!.	xck fqtokt!.
4	Operacao Invalida.