Exercícios TP de Técnicas Criptográficas

Manuel Barbosa at di.uminho.pt

CSSI – 2012/2013

Grupo I

Efectue a criptoanálise destes quatro criptogramas que foram obtidos recorrendo a cifras de substituição, *shift* e Vigenere. Apresente todo o trabalho que desenvolver, incluindo o código, e explique como chegou a uma solução. Não é suficiente apresentar o texto limpo.

```
EMGLOSUDCGDNCUSWYSFHNSFCYKDPUMLWGYICOXYSIPJCK
QPKUGKMGOLICGINCGACKSNISACYKZSCKXECJCKSHYSXCG
OIDPKZCNKSHICGIWYGKKGKGOLDSILKGOIUSIGLEDSPWZU
GFZCCNDGYYSFUSZCNXEOJNCGYEOWEUPXEZGACGNFGLKNS
ACIGOIYCKXCJUCIUZCFZCCNDGYYSFEUEKUZCSOCFZCCNC
IACZEJNCSHFZEJZEGMXCYHCJUMGKUCY

KCCPKBGUFDPHQTYAVINRRTMVGRKDNBVFDETDGILTXRGUD
DKOTFMBPVGEGLTGCKQRACQCWDNAWCRXIZAKFTLEWRPTYC
QKYVXCHKFTPONCQQRHJVAJUWETMCMSPKQDYHJVDAHCTRL
SVSKCGCZQQDZXGSFRLSWCWSJTBHAFSIASPRJAHKJRJUMV
GKMITZHFPDISPZLVLGWTFPLKKEBDPGCEBSHCTJRWXBAFS
PEZQNRWXCVYCGAONWDDKACKAWBBIKFTIOVKCGGHJVLNHI
FFSQESVYCLACNVRWBBIREPBBVFEXOSCDYGZWPFDTKFQIY
CWHJVLNHIQIBTKHJVNPIST

KQEREJEBCPPCJCRKIEACUZBKRVPKRBCIBQCARBJCVFCUP
KRIOFKPACUZQEPBKRXPEIIEABDKPBCPFCDCCAFIEABDKP
BCPFEQPKAZBKRHAIBKAPCCIBURCCDKDCCJCIDFUIXPAFF
ERBICZDFKABICBBENEFCUPJCVKABPCYDCCDPKBCOCPERK
IVKSCPICBRKIJPKABI

BNVSNSIHQCEELSSKKYERIFJKXUMBGYKAMQLJTYAVFBKVT
DVBPVVRJYYLAOKYMPQSCGDLFSRLLPROYGESEBUUALRWXM
MASAZLGLEDFJBZAVVPXWICGJXASCBYEHOSNMULKCEAHTQ
OKMFLEBKFXLRRFDTZXCIWBJSICBGAWDVYDHAVFJXZIBKC
GJIWEAHTTOEWTUHKRQVVRGZBXYIREMMASCSPBNLHJMBLR
FFJELHWEYLWISTFVVYFJCMHYUYRUFSFMGESIGRLWALSWM
NUHSIMYYITCCQPZSICEHBCCMZFEGVJYOCDEMMPGHVAAUM
ELCMOEHVLTIPSUYILVGFLMVWDVYDBTHFRAYISYSGKVSUU
HYHGGCKTMBLRX
```

Grupo II

Os seguintes vinte criptogramas foram obtidos recorrendo a uma cifra one-time-pad, com aritmética módulo 26, para cifrar duas mensagens diferentes. Ou seja:

- A chave é uma string do mesmo comprimento do texto limpo.
- A cifração é feita caracter-a-caracter.
- O criptograma é dado por $c_i = \mathsf{Char}((\mathsf{Code}(m_i) + \mathsf{Code}(k_i)) \pmod{26})$.
- O texto limpo é dado por $m_i = \mathsf{Char}((\mathsf{Code}(c_i) + 26 - \mathsf{Code}(k_i)) \pmod{26})$.

A utilização do one-time-pad é correcta, a menos da utilização de uma das chaves duas vezes e de um gerador aleatório fraco. O gerador aleatório utilizado para as chaves foi o nativo da linguagem Python, inicializado com a hora corrente. Identifique os criptogramas que utilizam a mesma chave e, se possível, tente recuperar (parcialmente) o texto limpo.

```
RHTINACZAWXPCQKCPHDMXSRFJAVVNWHZOAZESCQFKVQUPI
QFIDODBKTOKASLZRQCLYPENUOGRIDUJLWBXYBHVGXCLNRT
ULQWDCIEAMVXLEESIVFJTTOEJVCGFNAQJXSZQIYHDHNFWG
ZBDVSRWYNFYUJFSBFXVMFLBOCWHFWTPUQCGXMVKBLUCSXO
DJPCJWOMVEVIFAXDJLPPKEVFLDSHHSETLNEZHCSFHZQWGN
WBFDVHSDQINMTCOFARXTNELCKPUCKIKZQRSGFWTPZZKKQU
BQYLUAEOMNGBQF

EOIYXFDUATBLIFIVACYWXLILMCRBMZWHJHXDJYOEJQMDGW
JRGOKOVGNNNRGPYJKSZHRETCRLXDDYJVJHPKASRLPQMZGM
DOMFGVUQPIHVNYMANZXOJHQUHXYEHWEAKOCINYOEOOYCOL
BPYEPTIKWOADLHKAXMUEBHQDJEANJKBXCLFQNQIXYMCPKP
GJEBDYMLQWTBARFSFYKISJDADMGICKXREJJNJQDDZDWQOG
XKZYCGZDJCDXSEXTKPNPNJOGYIYVNEESAJSYKUCLTZMQIE
VPTXTEXXNAUNAD

SWNBDQVETCAXIIVJGYBTXHGPVWUHKAOULTSHFNADFONVVN
WOJLOPZMTNUYTOAUBKLACMSMCBHZGRSOAZRQTFDYDJHVEN
WHXLMKEZBQLCXIMVGJRERBMGAYSZZWVALMJSSJFOLUMSPL
KATYQKWKOVQKNIENGOYWMBWDBPODQOKZXTXXINCMAPCEEL
TOSSYTUULGANQRDLBJXJVQPZHCPTTWTOMUIPLRBHBKGAIY
YLOAMSKOHCIZENTYZGIBYFMNRIVTHJGNTQDHOPVAFHNERI
JNXOZIWQDOVFSY

SWZUZKDJMYOOCFELLFAJAOXSWDLWEVQBDQWXBQIVTTOQHE
KEBMGFKNHBHXYKSIIZSUQMJUXEOYDAXVAMRZITWXMIGCYQ
UHASJGRGIYDJJRUBHFJJAONULYILWRTDEMVIEDVYKTTZLE
QESHGEBCYAPXOVBPOUFIJDEZOUBDVVVMIKMSHSFBVFMRGZ
VYOOBWNLDTXWHZNQFYRTMDIGRTSMSILUFXCCLEBUQWPGJN
BBWNYBLSNELKXBZKKUNMKMHFAYCINMLVNWMSDYQAFRYEXF
PUOUZNFGBEMRTN
```

YAGCPYPZJSFJNWVLFKTYZKOUJZRRGVCAFHWDADMQLSNEBI
OXLREDCBJGNLYQWPVLYCMEMDAJGTKKQDUBMDBEIWIQERXB
XRBCCSWTBPRFUHASDTRMTGBRARYGIPXGRZQWCHSVXCRHGG
BNTMGIDWWOUANCCHUBPOQPYNPCIXQPFDRHGTVESYVRYHCJ
ZWOCBKHCXFOWGJERCNUHVBOHWLVWTPFHXFKWRIUNICQLDJ
OWCCEGAQHXHFQVTVFNAZSWKVBQRWLIQQHBJJGODDVLTBKG
LHMFBXHABAOKAH

SIPARKSRGJKBXODBSRCOTTKDLHQMBPTIZCNJFWGKCTRDBC
JONAGMTCIOHLAKBXZFUWOCYURUQINWXVKHCBGLITHIDZJZ
MXHDXLKOCMJSMWINDJTDPUCTHEZOQWBTNPIESOVKHQADUO
YKEUZKDKONRZZOKFOPHUVJVREBNHHJFNOHDABEPQSCAYPJ
CTXKZEBCGASVXJMIBCTYJEROQQZABVECYKEDAEXDHOHYJO
LKHNMNXVACQYFTIUVOFZLAOOLKHGNGSUMIEQZJSBKKXODR
KSSUEORINQKTGJ

ASWHCYPFDJQNALQLVSXLLROFXMFVUSLHSNWGLXFTTMINIO
QHUZUBZLXPKMFDLBKWAHZKUANIQGDMESUSNCSHXYHVTWLE
FQVKFYWZLROJNRTHHWWCNGZVZQAGRTXKMTHSYMHUBKMNQJ
OAKJWWHJERBOFDGPGDFUYTOGRUHRLGUMYATMUMOWMRYUSR
YVJTCMTEQQPYFYKJBUWYKKDFUJXIPVBKYFDCABVEABUBXO
WSOTRNXAHIZVQQRNHMQDZOATVNYEIHBOVMGAXZSSVRXENQ
HSESASXMVWWAHU

PYBLGOUOKKYKXGFKNNPBKFPSENGLKQENLQXNRENBSMOSEI
PNWMPYJHCHTOQVMQGMCSNUSLZFZMRMWGNGIGVZWEHUWLUU
HECNLWJLVJQSMRWMJOWIHTXOZLIPQEQFBIMRANSKCEGCUA
WECMYTSKAMVLXMXKZKPHITDICHGAZHJENYMUGDUBCKOFWJ
QKTPGBEFHDIHMDSMGVBCBVDTTYIYOZITYMEYASPAHRWJKI
GMOMFBVBIZECZDUTRVZMIZVAGMWGGWKESBMLZFCHQNUTEL
XGGEFDDDSLLPGD

EZHDUMIRNCCTIPFZFYOCJYYOLPGOTDUVYPAIRJZFPLZYQS
XXWSJUDBEPRYKTMMXVSPSCTBAIBQPRGQZYIJDDYJYDTDSY
XIBQBJMEZVVNAMDLXSVAMTXQCJSMLKOXXJNVYXIZXRSDKL
MUMCHNAKAYZQJDQMACMSBNFJJMUDTLDSIOCDJORLUWZTHJ
PNSLEVDERRJQZCCKHPPUVIUOVKCWUGJLOOBUEWGEEWBPRR
XOZMWFNBTOHWHGLJHEKHOHNPZVBGSYKHQEJUVFWPURWJNA
SJGFEBSJWIFHCB

ZCIQKYYDJLJOGVIZTYFVHPLJZOTBUJZOSWMPUYSRDOBEBV
QUFDSCJGECQVNKNSGSASGNZEDKAYJQSRQVPAVAALZBCQNZ
UTNDTDLKAZSHVHCPFUINNAGUVWVNXKSSNYWYFJMQMTBAQU
LTYRQZDZXZIVAOAPXGGHTBGEDFKLDCOLKYKOOGFVRYMCDC
HFNPAZBIRRMDXRXMOAWPIINGLTAFBAFXRNGRWYQEBVPGZI
EEXTKNJMYTFERQGNAYOKVUNIBJHBCUUMAUPCNXORTIKDRR
BKFPVTDXJAWUAM

MRPLDJCGUZVVOBZCOBZRNGDGCDCCVLBLQVENFOROGCZDIS
ERHVIRELKSUAKYUEDQFQHMVAKKYPVXLYIMGZYFTBNXWQRV
NFZDIWIXHQNKVPCMIQEMTJYQEETIZRBEBZNDPBEXTSRHDH
HMJCWYKEKWYPTCCJUYBXQFFULCHZXORFGHKWBKOZHHWOAB
AEXYOTRLWVHJNDJMFIWGGPVUHRSTRDXRHNHJGXMPHVHUOI
KUOJRGFMEYCWCZAQWSNXJCNBREUKVMTZHGERQHGSPXECOH
WJAGASXDRURMLU

DENHAXEKWUVMXEZOYMKPYPRUZOPCGDZIVNVJFBIDKACVJD
OLMVXQHPDCQBBELRNIKEBYEIPHYLDVRUHVVGEKNSEPQOEF
LUXSCSRKUJGGKYGHAJWSKPDENXFDDRAHJLRBGAJVPQVPKO
EXNXRDMFWIBOBYYZFHHMTFJDAWLIRNADLOZFFAYGVIEKDM
SKRCKUUHYGMYPLQMJYHMXQIGYOIWIHFVNKIRQUSIKMUXGY
RLFGVNGUWUHAFKUVFGPTIGTGKASPDHTZPGAUJDDMFDSFOO
DPJDHPHXNCXXEB

VRSMEBAOLNHMWWLIDTXOLUDSCKWUDFUBLTZQDQPISBBUQF
VHTBGNIKHHKZEDGAYAOVDNFJFGIJAGPSXBZSBWDVHBNPQX
PZUGYSFHUZKEURTISEPBCSMUERYECDPFJIULNWEESFMHJO
XNVCBXDUZSPGKZPBXPMISZZITNXWWGDIFAXOVGLQAQBPWL
QHDCHOKFCVJOMHWWSSANQOVJSMIDAQPSFOCIULLRDBSUFT
OYSVRAXUQQZRYABGIOLIKXLLTIICGOWMPLNNPZNZZIIIQI
NPFCXTIOHXHQVR

OHBWHLTNCVVFQZUXYJYGMTCUCXTYXWXLFKYLQYGYIALTGP
CUOPBABOMALBYECNCSAAXVAEHGPGIXPKMPRZHWPLONPJZN
IQSPOSWSSXWKKFGQJJTYZAKWPBIXQZMDXVKRQYSOULZRPX
WGYSRLSXQMSQFXWZKQMKVAPRMTKLHXXPUSJAOCLSUZPZUI
FSDNTSDCKZJQIWEHYWDPLPQTGXAGZRSRVXQNOGJMXIVDQA
DKTPLPWYBNEIDSUEEMXEGDXKXBHCLKHSTHIAMCAPUVDEWU
DYQNYYZTWPSETG

IYXDJNGDMTBEJWOCBEPNRJYEJMZVBRDGTETVPGZEWWMGJQ
UHYBOMRJPNMUBSVPYMDBKAWQCMKUPJPUJYYFCSSYDLDRRW
AYMKNJUGOLPXWYPRVEKLOECVNNHABYQOXAKXXBJASMNKSM
MTDKNFLAPLJNLWVGVKQJQQPQIQYOGZNCHGZZXHEDSRCTKQ
BGCKJEBHFUCPDSJBIJQELPFVDQCANYNOSECQVHDHSWKBDF
YGHQMEIEDERRVKLTMIEZYRMYJEKBNWWZLJGQNSVQTEZPYJ
USMIQXMIIWXFVT

HKZIYOSBMHPOHOWMRHTIENSCHJQNZQEFELQAXTUAFGXCLG
ESEMHVPRNYEACFGUNZIQJENGYIYWMEOXSJLWBSWWFOSVSC
NDGAFAZECWQZBYGGXKEYTRWOMNWXLXHMOVYWMNSRTJKOZF
OOUCBTIQJEXFDWVBCCDXDSPFBJVNTYGVGKBMYBXOIZYEIY
FZGLBIWNBHUPHYOBSEAVQIHFCBQRPWMBLMQBKDVQZIBNYJ
YEZRMLJMAKCGMHALBOSOETRAGBZWFXKDRYRSLQMDDRRMOJ
KLOEGOYWZCYOST

FAKIWAHNUASFHVLAUSJIPLWPNAARZMCFPWFKNNASMNLETG
PSLOOLQNVDJWEIHOMBCOEVXIVIKTUPLDQNCTQGMBDEDZSH
MZNLSFJNPWFLFDUMJINCMMBROYHOMRUMNYBOWUGCGTEGYU
JCQCCVBMQRYFIIDIKRXSKKWDIUEAWIYJSFKEFGLAYVVDEY
XQQQTQBBDPXCFHKGMQZZSNLBEZGGUGFLHYSDXVJPADEWBA
CJBJZSCJEUDPXZDWJEAOVPJGMSREXWCTEXLXJOSEPZIVPP
KJPFNXATQQNBGP

FEEDIEYEBTGTOWYBUNSDSFHMFMFNWLQUKSDSAVCGTBRZZM
EOWFIPGYMJDFTSFGTBPVMEPOAXJZNBLZUWNRFGWWHEPUTN
KQXFQFGSHQOSXPYIWXUVXVXEXIYYMRCEIAEPCZTDCWRHYC
LYTUOGPJNHETRJOYNEWTWCAHVEUFGUNPHOZWSWULAJUVAL
BEQVUWTGUXUBGUCMBQNEWSMZVMTYTLHDTZKTAGRJANZYXI
YKNMEQMMRAQLCUJNPFFYBSTDQDOEROEFBJZEFWMPOJBMZX
RWOLKJWLGAJNCL

IBRCYIAIRODCKCOROCVCRSCCDJTTZNRMYREQPIDBRQKBTQ
GYFWCAROIRRUBWSLOBBAUZMBARUYVWWHLSIGCOTFZLQINZ
LGXFDAMENDYLZYBQYRQFTAHGEWGBNCXDHNBOLMUJAPBPFG
RQRGUOXJZMMULJHDFXXECHKAJRRXRWNVTVNLUNBBALWLRQ
TJASNEMLBJVXKRVUVNUFEAVDNQPUXSYXTFNJGXZJVTLDWX
BBIRLYITHKKZESVGGVYYXPPRJDPEJMDVYFWRWVRUXXYGLI
YOTYMBENSPJVDY

WVSAKLLAVNGTNLMEBAWXHGQIQSYDGKVBLBSVZSMLJSWBIV
YWTUBSPOHQPLOLDWIVPMOYWSDCXYXYOOYCOUGEWLTFZRJE
ZASOGPTRYNCLYVQGEXARHHRSPZVBNAQYDXHFFZCWYECFAN
OAHWQCDSMNUJNFHKJXHENQFZZFWRLQHUSWTBUBMOMEEWBP
CQJFUBWYDVMIZWNOPDBTNMRWUBTNOVKIFJFBJJBDAEZWWJ
WVOCUPBTRKTDINVPDUHIBSDYLOVXRZYLFFWPNVBNAMKQRC
TDYDHIUYWQOSVQ

Grupo III

O modo de funcionamento de cifras por blocos Electronic Code Book não esconde todos os padrões no texto limpo. Pretende-se demonstrar este facto da seguinte forma:

- Escolher como ficheiro de texto limpo um ficheiro de imagem que armazene os gráficos como uma matriz de pontos, sem compressão.
- Cifrar a parte do ficheiro que corresponde à imagem e criar um novo ficheiro de imagem com esse conteúdo.
- Observar na imagem que contém o criptograma que certos padrões da mensagem original são visíveis.

Grupo IV

O esquema CBC-MAC apresentado nas aulas teóricas permite autenticar mensagens de tamanho pré acordado $\ell(n) \cdot n$. Define-se da seguinte forma.

Seja $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ uma função pseudo-aleatória e $\ell(n)$.

Algorithm $\mathsf{Gen}(1^n)$
$\mathsf{k} \leftarrow_{\$} \{0,1\}^n$
Return k

Algorithm $\mathsf{Ver_k}(\mathsf{m}, \mathsf{t})$
If $|\mathsf{m}| \neq \ell(n) \cdot n$ Return F
Return $\mathsf{Mac_k}(\mathsf{m}) = t$

Algorithm $\mathsf{Mac_k}(\mathsf{m})$
If $|\mathsf{m}| \neq \ell(n) \cdot n$ Return $\perp$
$(\mathsf{m}'_1, \ldots, \mathsf{m}'_{\ell(n)}) \leftarrow \mathsf{partition}(\mathsf{m}', n)$
$\mathsf{t}_0 \leftarrow 0^n$
For $i \in [1 \ldots \ell(n)]$
    $\mathsf{t}_i \leftarrow F_{\mathsf{k}}(\mathsf{t}_{i-1} \oplus \mathsf{m}'_i)$
Return $\mathsf{t}_{\ell(n)}$

partition parte a mensagem em blocos de tamanho n.

A diferença para o modo de cifra CBC é que este esquema fixa o IV a um valor pré-determinado e retorna apenas um bloco como *tag*. Estas condições são necessárias à segurança. Em particular, utilizar um IV aleatório ou retornar todos os blocos levam a um MAC inseguro.

Pretende-se neste exercício que se demonstre na prática os ataques que são possíveis ao CBC MAC quando é utilizado um IV aleatório ou quando são retornados todos os blocos processados. Ou seja, pretende-se uma implementação destas versões inseguras do protocolo, bem como de um algoritmo que falsifique MACs com sucesso.

Grupo V

1. Resolva o seguinte sistema de congruências:

$$\begin{cases} x \equiv 12 \pmod{25} \\ x \equiv 9 \pmod{26} \\ x \equiv 23 \pmod{27} \end{cases}$$

2. Resolva o seguinte sistema de congruências:

$$\begin{cases} 13x \equiv 4 \pmod{99} \\ 15x \equiv 56 \pmod{101} \end{cases}$$

3. Utilizando a técnica RSA foram obtidos os seguintes criptogramas. Os parâmetros públicos utilizados foram n=31313 e b=4913, no primeiro caso, e n=18923 e b=1261, no segundo caso. "Quebre" estas cifras, tendo em conta que, devido à dimensão de n nos dois casos, o problema da factorização do módulo é relativamente fácil de resolver. Tenha ainda em consideração os seguintes pontos:

   - Para recuperar o texto limpo necessita de saber como foi codificado. Neste caso, cada letra é tratada como um elemento de $\mathbb{Z}_{26}$, e cada trio de letras $L_1L_2L_3$ é codificado num número $L_1 * 26^2 + L_2 * 26 + L_3$.
   - Escreva e use os seus próprios programas para desempenhar esta tarefa.
   - Submeta listagens comentadas das partes relevantes do código que produzir, o texto limpo que recuperou e uma explicação de como procedeu para resolver o problema.

6340	8309	14010	8936	27358	25023	16481	25809
23614	7135	24996	30590	27570	26486	30388	9395
27584	14999	4517	12146	29421	26439	1606	17881
25774	7647	23901	7372	25774	18436	12056	13547
7908	8635	2149	1908	22076	7372	8686	1304
4082	11803	5314	107	7359	22470	7372	22827
15698	30317	4685	14696	30388	8671	29956	15705
1417	26905	25809	28347	26277	7897	20240	21519
12437	1108	27106	18743	24144	10685	25234	30155
23005	8267	9917	7994	9694	2149	10042	27705
15930	29748	8635	23645	11738	24591	20240	27212
27486	9741	2149	29329	2149	5501	14015	30155
18154	22319	27705	20321	23254	13624	3249	5443
2149	16975	16087	14600	27705	19386	7325	26277
19554	23614	7553	4734	8091	23973	14015	107
3183	17347	25234	4595	21498	6360	19837	8463
6000	31280	29413	2066	369	23204	8425	7792
25973	4477	30989					

12423	11524	7243	7459	14303	6127	10964	16399
9792	13629	14407	18817	18830	13556	3159	16647
5300	13951	81	8986	8007	13167	10022	17213
2264	961	17459	4101	2999	14569	17183	15827
12693	9553	18194	3830	2664	13998	12501	18873
12161	13071	16900	7233	8270	17086	9792	14266
13236	5300	13951	8850	12129	6091	18110	3332
15061	12347	7817	7946	11675	13924	13892	18031
2620	6276	8500	201	8850	11178	16477	10161
3533	13842	7537	12259	18110	44	2364	15570
3460	9886	8687	4481	11231	7547	11383	17910
12867	13203	5102	4742	5053	15407	2976	9330
12192	56	2471	15334	841	13995	17592	13297
2430	9741	11675	424	6686	738	13874	8168
7913	6246	14301	1144	9056	15967	7328	13203
796	195	9872	16979	15404	14130	9105	2001
9792	14251	1498	11296	1105	4502	16979	1105
56	4118	11302	5988	3363	15827	6928	4191
4277	10617	874	13211	11821	3090	18110	44
2364	15570	3460	9886	9988	3798	1158	9872
16979	15404	6127	9872	3652	14838	7437	2540
1367	2512	14407	5053	1521	297	10935	17137
2186	9433	13293	7555	13618	13000	6490	5310
18676	4782	11374	446	4165	11634	3846	14611
2364	6789	11634	4493	4063	4576	17955	7965
11748	14616	11453	17666	925	56	4118	18031
9522	14838	7437	3880	11476	8305	5102	2999
18628	14326	9175	9061	650	18110	8720	15404
2951	722	15334	841	15610	2443	11056	2186

4. Escreva um programa para calcular símbolos de Jacobi utilizando as seguintes propriedades.

   (a) Se n é um inteiro impar, e $m_1 \equiv m_2 \pmod{n}$, então
   $\left(\frac{m_1}{n}\right) = \left(\frac{m_2}{n}\right)$.

   (b) Se n é um inteiro impar, então
   $$\left(\frac{2}{n}\right) = \begin{cases} 1 & \textit{se } n \equiv \pm 1 \pmod 8 \\ -1 & \textit{se } n \equiv \pm 3 \pmod 8 \end{cases}.$$

   (c) Se n é um inteiro impar, então
   $\left(\frac{m_1 m_2}{n}\right) = \left(\frac{m_1}{n}\right)\left(\frac{m_2}{n}\right)$
   e, em particular, se $m = 2^k * t$
   $\left(\frac{m}{n}\right) = \left(\frac{2}{n}\right)^k \left(\frac{t}{n}\right)$.

   (d) Se m e n são inteiros impares, então
   $$\left(\frac{m}{n}\right) = \begin{cases} -\left(\frac{n}{m}\right) & \textit{se } m \equiv n \equiv 3 \pmod 4 \\ \left(\frac{n}{m}\right) & \textit{nos restantes casos} \end{cases}.$$

   O programa não deverá fazer factorização mais complexa do que a divisão por potências de 2. Teste o seu programa para calcular:

$\left(\frac{610}{987}\right)$, $\left(\frac{20694}{1987}\right)$ e $\left(\frac{1234567}{11111111}\right)$.

5. Para n = 837, 851 e 1189 encontre as bases b, para as quais n é um pseudo-primo de Euler. (Nota: n é um pseudo-primo de Euler na base b se o símbolo de Jacobi $\left(\frac{b}{n}\right)$ é igual a $b^{(n-1)/2} \pmod{n}$.