Report Project 2 -
William Lewin & Elias Chahine
wlewin@kth.se, echahine@kth.se

Summary

Task

Professor Alice is sending a problem to the student Bob according to the procedure in section 1.1.1. You are supposed to solve Bob's problem. However, you need to start by cracking the cipher sent to Bob. When translating to ASCII, you can assume the base 256.

The numbers is the list seems to be of the same size. Why?
Motivate your selected mathematical model!

Task derived from previous task:

Congratulations! You have now managed to crack the RSA cipher. Your task is as follows: write a method R SAcrack[cipher, n, e] that w "ill crack a standard RSA cipher and delivers clear text from the string cipher . When you are finished with your method, you should " investigate how long it will take to crack the ciph er of the Swedish text DISKRET MATTE, DE E FETT NAIS! for different sizes on y our public key n (100–200 bits). Visualize your results in a proper graph. It is very important that you study the section 2.3 in the instructions. Your graph should lead you to a model where you can predict how long it would take to crack a cipher if n is 1024, 2048 bits or 4096 bits with your computer.

1.1.1 Using RSA for Authentication

Result

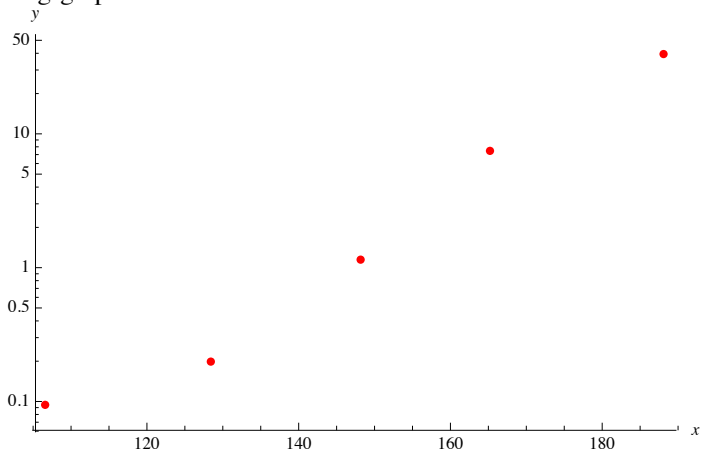
The numbers in the cipher are roughly of the same size because the message needs to be broken down in to smaller pieces (smaller than n) in order for the cracking method to return a unambiguous result.

After cracking the initial RSA puzzle, we were introduced to a new task. This task involved writing a method for decrypting a cipher and calculating the time required to crack the cipher depending on the size of the size of the public key (N).
The time required to crack the cipher will strongly depend on the size of N, seeing as the main problem of cracking an RSA cipher is the prime factoring required to reverse the encryption process.

Approx N.	Processing time
$2^{100-110}$	0.087013
$2^{120-130}$	0.218892
$2^{140-150}$	1.26341
$2^{160-170}$	7.85949
$2^{180-190}$	41.0367

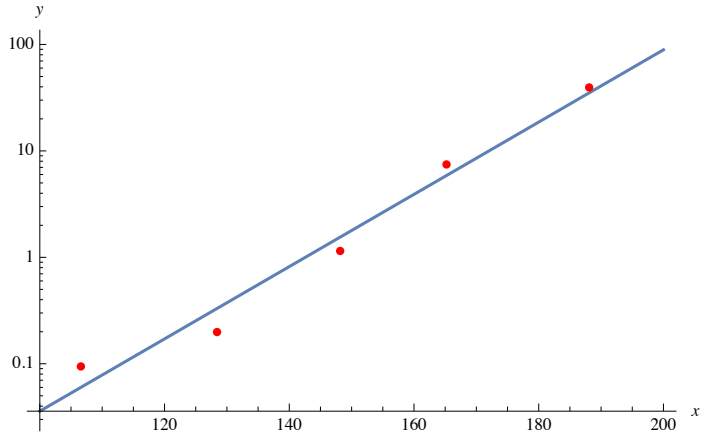
Our mathematical model of choosing was a power function. As the graph below shows, when plotted as a standard graph, the values show an exponential curve upwards.

Log-graph



When plotted as a log-log graph, the values are displayed as a straight line. We chose this model based on "If data fits a straight line in a log(y)-log(x) plot, then the model is a power function."

Log-lin and function graph



Based on this model and the data we have gathered, a prediction of the running time of larger N was created.

N	Estimated processing time
2^{1024}	8.46925×10^{29}
2^{2048}	4.97244×10^{64}
2^{4096}	1.71403×10^{134}

Code

Part 1 - Alice & Bob

In[20]:=
```
nAlice =
  14 625 052 655 936 968 690 439 555 839 903 578 410 656 653 152 402 266 773 966 305 647 313;
eAlice = 9103;
```

In[22]:=
```
nBob =
  891 257 371 165 353 198 597 686 733 461 449 215 081 540 410 212 576 138 836 198 818 353;
eBob = 6389;
```

In[24]:=
```
cipher =
 {4 430 210 573 981 968 733 755 487 447 818 174 029 880 701 596 884 938 356 564 639 003 222,
  2 851 153 559 114 750 987 221 731 555 966 122 250 132 614 620 940 651 980 091 426 872 783,
  7 706 806 242 317 911 603 638 504 323 916 964 668 918 097 951 180 075 641 184 510 131 492,
  3 684 227 409 638 269 668 959 612 918 228 029 487 936 911 649 602 484 180 495 870 360 134,
  118 390 233 898 339 944 208 891 887 229 177 617 429 671 446 231 727 377 499 079 969 939,
  11 722 755 771 961 643 408 261 179 942 790 037 153 532 726 213 028 762 901 063 706 465 880,
  13 859 812 858 423 457 720 552 211 210 177 480 158 021 647 193 707 898 449 643 563 546 715,
  8 458 200 553 283 608 142 592 103 589 289 032 864 817 455 387 331 640 007 234 518 389 001,
  4 203 062 251 335 458 332 458 951 560 795 586 742 826 858 693 125 368 654 700 629 553 235,
  11 246 017 592 897 572 378 455 633 552 556 960 091 533 916 541 949 272 900 160 606 355 164,
  528 729 157 482 321 024 630 053 104 368 111 400 799 635 858 352 035 984 875 544 277 541,
  14 149 609 332 514 930 578 966 644 135 292 557 372 570 883 055 216 115 495 760 639 164 502,
  6 806 693 065 526 086 081 574 819 023 329 054 541 636 484 663 342 321 332 752 294 055 806,
  10 173 212 145 631 688 548 727 430 811 241 580 523 813 551 542 356 631 310 237 779 750 220,
  6 759 760 040 422 316 973 322 423 168 431 166 905 944 837 651 248 859 901 210 533 615 722,
  11 488 644 301 666 039 406 597 033 276 777 701 689 096 765 993 957 684 011 886 918 061 630,
  10 421 162 414 768 710 251 337 302 689 028 405 059 375 889 881 305 351 342 596 162 625 647,
  9 496 359 638 348 947 223 087 997 149 242 581 766 202 717 301 485 526 573 539 185 466 328,
  6 021 527 813 042 783 330 194 131 859 432 644 092 333 420 618 174 656 828 129 608 651 886,
  1 862 712 526 928 231 860 818 857 411 345 402 991 522 792 361 353 775 067 366 437 910 859,
  5 770 870 235 309 404 532 139 302 393 417 410 250 573 907 045 700 501 009 827 409 811 147,
  7 754 796 273 877 044 686 482 387 739 917 927 617 087 893 337 176 831 108 280 229 804 774,
  1 689 445 149 306 311 402 232 833 597 026 076 887 079 752 228 378 923 964 144 759 423 086,
  1 448 530 945 006 829 418 385 610 770 415 140 752 515 002 298 435 493 622 427 431 846 992,
  7 445 869 507 503 892 181 715 468 267 483 205 548 161 014 207 123 217 105 479 157 092 169,
  10 975 830 923 316 626 122 520 530 596 740 793 665 823 967 057 824 410 678 091 427 215 458,
  9 807 539 864 876 132 360 289 770 201 821 020 858 873 102 738 795 986 884 424 480 971 219};
```

In[25]:=
```
qAlice = 3 086 973 200 587 318 781 402 218 709 909 851;
pAlice = 4 737 667 516 243 564 271 342 073 671 581 763;
pBob = 457 792 861 431 456 494 922 119 050 690 577;
qBob = 1 946 857 293 446 017 662 241 143 108 141 089;
phiBob = (pBob - 1) * (qBob - 1);
```

In[30]:=
```
c1 = PowerMod[cipher, eAlice, nAlice]
dBob = PowerMod[eBob, -1, phiBob];
```

Out[30]=
```
{444 970 409 279 297 770 090 044 313 053 201 614 952 097 369 301 321 865 917 424 641 258,
 183 674 113 104 017 659 567 913 355 845 229 971 423 316 056 372 265 070 315 906 622 235,
 617 562 464 331 006 011 907 353 184 885 542 978 809 945 586 185 511 937 486 448 993 081,
 692 990 377 295 446 436 150 600 101 814 271 370 870 058 687 365 945 680 092 195 122 174,
 15 829 748 288 796 827 376 895 234 161 254 693 491 217 089 543 931 177 859 749 489 351,
 829 040 669 128 475 076 848 018 289 419 857 381 658 084 508 110 712 186 672 486 418 727,
 798 784 110 186 848 820 585 991 906 620 519 315 059 619 601 235 652 455 703 626 882 470,
 399 074 729 448 818 803 154 154 587 362 403 712 935 004 945 380 245 199 874 499 152 222,
 117 885 647 490 473 600 604 299 228 604 398 634 086 600 112 086 102 442 640 882 667 648,
 558 787 628 279 681 901 598 087 640 588 483 929 735 685 558 225 150 887 948 892 510 920,
 308 098 398 607 398 025 511 582 944 888 788 009 237 158 897 308 762 960 032 967 624 060,
 246 032 490 633 541 021 297 680 837 804 144 220 001 633 329 118 178 930 732 181 456 343,
 553 148 258 011 717 024 187 532 272 757 027 196 386 133 700 023 765 934 740 879 242 061,
 15 757 977 304 317 592 988 100 905 265 851 778 837 698 855 049 986 566 247 100 042 111,
 48 555 750 069 717 983 165 419 507 244 417 369 421 745 602 752 746 641 594 676 113 574,
 645 816 746 709 820 531 252 882 158 812 595 825 630 657 287 083 837 162 388 102 093 670,
 579 143 280 819 432 400 927 730 228 652 650 181 095 648 756 820 602 602 216 165 263 215,
 330 396 900 456 864 364 791 120 284 078 288 409 962 248 134 453 101 589 867 285 929 222,
 30 285 107 152 858 671 427 605 286 391 722 289 914 017 102 065 648 398 433 550 314 482,
 394 905 566 235 612 664 675 691 317 209 814 782 624 147 170 337 593 067 979 209 081 310,
 755 643 239 531 999 740 019 455 828 836 426 484 399 589 590 914 276 018 760 019 795 860,
 271 578 869 017 028 208 773 948 310 614 382 084 613 277 253 285 953 238 316 507 795 669,
 32 235 777 068 746 143 161 753 960 770 773 082 449 807 496 888 435 064 114 936 879 538,
 345 365 485 638 270 886 035 459 127 757 587 129 598 779 123 419 559 816 170 941 440 724,
 491 321 654 189 544 745 328 056 390 015 517 259 287 338 718 673 949 903 009 759 789 721,
 654 607 901 613 345 323 795 924 172 006 966 197 409 745 426 827 819 740 917 539 895 273,
 400 520 866 556 330 100 012 684 699 137 119 966 708 411 863 335 636 746 903 373 549 398}
```

In[32]:=
```
B = 256
For[i = 0, i ≤  Length[c1], i++,
 messageFrom = PowerMod[c1[[i]], dBob, nBob];
 q = messageFrom; ascii = {};
 While[q ≠ 0,
   AppendTo[ascii, Mod[q, B]];
   q = Quotient[q, B];
  ] ×
 ascii;
 Print[FromCharacterCode[ascii]]]
```

Out[32]=
```
256
```

Congratulations! You have now managed to crack the RSA cipher. Your task is as follows: write a method RSAcrack[cipher,n,e] that will crack a standard RSA cipher and delivers clear text from the string cipher. When you are finished with your method, you should investigate how long it will take to crack the cipher of the Swedish text DISKRET MATTE, DE E FETT NAIS! for different sizes on your public key n (100-200 bits). Visualize your results in a proper graph. It is very important that you study the section 2.3 in the instructions. Your graph should lead you to a model where you can predict how long it would take to crack a cipher if n is 1024, 2048 bits or 4096 bits with your computer.

Part 2 - RSAcrack

In[34]:=
```
ClearAll["`*"]
rsaCracker[cipher_, n_, e_] := Module[{storeAscii, messageFromSomeone,
    q, ascii, storePrime, pSomebody, qSomebody, phiSomebody
    , dSomebody, x, decrypt, storeCrypto},
  Array[storePrime, {1, 2}];
  Array[storeAscii, Length[cipher]];
  Array[storeCrypto, Length[cipher]];
  storePrime = FactorInteger[n];
  pSomebody = First[First[storePrime]];
  qSomebody = First[Last[storePrime]];
  phiSomebody = (pSomebody - 1) * (qSomebody - 1);
  dSomebody = PowerMod[e, -1, phiSomebody];
  For[x = 1, x ≤ Length[cipher], x++,
   decrypt = PowerMod[cipher[[x]], dSomebody, n];
   (*Print[FromCharacterCode[Mod[PowerMod[cipher[[x]],dSomebody,n],256]]*)
  ]
  (*Print[FromCharacterCode[PowerMod[cipher[x],dSomebody,n]]];*)
 ]
```

In[36]:=
```
n1 = 5 578 181 019 009 693 605 300 014 200 069 239 612 723;
e1 = 2789;
c1 = {1 608 192 936 300 849 447 357 116 938 533 648 061 107,
3 368 101 354 959 964 882 647 605 580 669 968 489 491,
2 067 852 446 080 194 085 197 089 058 618 838 874 024,
1 970 950 724 178 345 329 211 485 215 132 189 129 759,
947 134 178 411 813 608 445 435 420 328 616 718 804,
1 925 576 565 744 010 767 772 217 211 256 082 820 941,
4 605 459 822 148 021 013 012 052 952 352 972 584 253,
4 186 841 003 888 572 996 495 111 346 409 155 824 035,
1 846 041 963 789 369 070 974 811 225 481 636 796 431,
5 096 044 390 444 446 125 352 119 889 852 571 089 057,
4 605 459 822 148 021 013 012 052 952 352 972 584 253,
4 605 459 822 148 021 013 012 052 952 352 972 584 253,
1 925 576 565 744 010 767 772 217 211 256 082 820 941,
5 115 617 652 969 099 595 819 651 211 522 229 560 263,
4 186 841 003 888 572 996 495 111 346 409 155 824 035,
1 608 192 936 300 849 447 357 116 938 533 648 061 107,
1 925 576 565 744 010 767 772 217 211 256 082 820 941,
4 186 841 003 888 572 996 495 111 346 409 155 824 035,
1 925 576 565 744 010 767 772 217 211 256 082 820 941,
4 186 841 003 888 572 996 495 111 346 409 155 824 035,
2 041 435 395 349 385 173 865 782 389 645 455 193 305,
1 925 576 565 744 010 767 772 217 211 256 082 820 941,
4 605 459 822 148 021 013 012 052 952 352 972 584 253,
4 605 459 822 148 021 013 012 052 952 352 972 584 253,
4 186 841 003 888 572 996 495 111 346 409 155 824 035,
5 204 069 424 414 112 663 197 882 625 268 745 616 409,
5 096 044 390 444 446 125 352 119 889 852 571 089 057,
3 368 101 354 959 964 882 647 605 580 669 968 489 491,
2 067 852 446 080 194 085 197 089 058 618 838 874 024,
2 986 416 525 605 309 900 776 533 196 503 168 288 929};
t1 = Timing[rsaCracker[c1, n1, e1]]
```

Out[39]=
```
{0.306832, Null}
```

≈2^100-110

In[40]:=
```
n2 = 121 914 658 243 735 402 103 595 568 170 551;
e2 = 63 619;
c2 = {72 223 437 142 745 922 761 712 779 801 658,
33 340 725 534 051 141 497 601 756 886 800,
32 221 412 764 008 165 092 025 682 787 575,
120 681 464 919 026 052 203 103 890 611 480,
59 645 867 002 865 201 046 088 441 793 664,
93 146 411 292 283 398 968 522 997 511 128,
44 754 172 949 140 988 385 706 024 419 890,
48 539 000 215 501 430 549 676 716 832 222,
38 888 168 873 414 911 218 290 508 562 342,
24 945 388 829 311 567 639 254 635 725 552,
44 754 172 949 140 988 385 706 024 419 890,
44 754 172 949 140 988 385 706 024 419 890,
93 146 411 292 283 398 968 522 997 511 128,
25 914 381 598 839 031 224 009 077 432 647,
48 539 000 215 501 430 549 676 716 832 222,
72 223 437 142 745 922 761 712 779 801 658,
93 146 411 292 283 398 968 522 997 511 128,
48 539 000 215 501 430 549 676 716 832 222,
93 146 411 292 283 398 968 522 997 511 128,
48 539 000 215 501 430 549 676 716 832 222,
66 267 839 097 823 468 335 752 547 081 914,
93 146 411 292 283 398 968 522 997 511 128,
44 754 172 949 140 988 385 706 024 419 890,
44 754 172 949 140 988 385 706 024 419 890,
48 539 000 215 501 430 549 676 716 832 222,
66 135 258 968 373 472 547 906 451 453 888,
24 945 388 829 311 567 639 254 635 725 552,
33 340 725 534 051 141 497 601 756 886 800,
32 221 412 764 008 165 092 025 682 787 575,
115 613 677 863 339 485 569 819 355 518 945};
t2 = Timing[rsaCracker[c2, n2, e2]]
```

Out[43]=
```
{0.094384, Null}
```

≈2^120-130

In[44]:=
```
n3 = 455 655 141 456 960 940 308 510 806 241 646 658 819;
e3 = 77 513;
c3 = {229 856 101 785 498 353 633 365 686 324 091 888 062,
6 898 543 326 360 707 900 578 290 633 740 841 130,
155 352 807 959 041 102 282 886 683 051 662 684 796,
314 893 222 850 282 505 783 380 209 933 883 265 562,
70 114 904 766 560 545 494 270 380 740 143 622 926,
179 416 050 484 005 634 773 636 110 729 539 841 447,
259 382 091 522 021 082 769 104 400 203 500 074 135,
86 203 446 659 210 353 812 339 302 068 440 599 223,
247 369 200 372 479 339 628 745 849 133 547 768 022,
231 551 185 461 132 829 182 195 918 270 777 623 595,
259 382 091 522 021 082 769 104 400 203 500 074 135,
259 382 091 522 021 082 769 104 400 203 500 074 135,
179 416 050 484 005 634 773 636 110 729 539 841 447,
142 560 303 329 889 017 622 909 380 182 692 309 072,
86 203 446 659 210 353 812 339 302 068 440 599 223,
229 856 101 785 498 353 633 365 686 324 091 888 062,
179 416 050 484 005 634 773 636 110 729 539 841 447,
86 203 446 659 210 353 812 339 302 068 440 599 223,
179 416 050 484 005 634 773 636 110 729 539 841 447,
86 203 446 659 210 353 812 339 302 068 440 599 223,
185 874 809 763 790 862 698 014 047 508 581 899 317,
179 416 050 484 005 634 773 636 110 729 539 841 447,
259 382 091 522 021 082 769 104 400 203 500 074 135,
259 382 091 522 021 082 769 104 400 203 500 074 135,
86 203 446 659 210 353 812 339 302 068 440 599 223,
170 245 136 740 506 555 122 701 271 711 162 938 767,
231 551 185 461 132 829 182 195 918 270 777 623 595,
6 898 543 326 360 707 900 578 290 633 740 841 130,
155 352 807 959 041 102 282 886 683 051 662 684 796,
310 110 795 247 312 160 025 167 303 315 805 902 060};
t3 = Timing[rsaCracker[c3, n3, e3]]
```

Out[47]=
```
{0.198565, Null}
```

≈2^140-150

In[48]:=
```
n4 = 399 631 185 399 565 082 910 863 599 325 056 141 232 359 753;
e4 = 38 131;
c4 = {178 403 006 527 178 050 571 555 949 039 536 217 650 435 227,
382 333 861 151 321 333 498 960 151 740 199 072 902 842 743,
1 761 584 675 590 839 734 287 976 390 632 540 296 552 968,
22 404 752 690 349 690 053 084 894 825 719 736 337 297 688,
165 821 407 921 528 136 252 805 764 703 861 101 446 997 623,
240 364 220 141 028 992 626 632 272 034 750 921 311 523 065,
302 590 304 340 353 376 349 353 538 444 305 692 106 774 979,
101 560 557 675 965 682 301 030 257 770 383 367 178 916 685,
62 888 716 702 837 907 792 667 650 514 263 321 420 354 985,
128 032 200 604 782 244 713 434 477 043 518 271 901 000 964,
302 590 304 340 353 376 349 353 538 444 305 692 106 774 979,
302 590 304 340 353 376 349 353 538 444 305 692 106 774 979,
240 364 220 141 028 992 626 632 272 034 750 921 311 523 065,
367 975 550 705 439 116 208 504 452 788 241 548 070 322 850,
101 560 557 675 965 682 301 030 257 770 383 367 178 916 685,
178 403 006 527 178 050 571 555 949 039 536 217 650 435 227,
240 364 220 141 028 992 626 632 272 034 750 921 311 523 065,
101 560 557 675 965 682 301 030 257 770 383 367 178 916 685,
240 364 220 141 028 992 626 632 272 034 750 921 311 523 065,
101 560 557 675 965 682 301 030 257 770 383 367 178 916 685,
293 586 670 426 676 443 697 591 348 400 786 941 156 834 334,
240 364 220 141 028 992 626 632 272 034 750 921 311 523 065,
302 590 304 340 353 376 349 353 538 444 305 692 106 774 979,
302 590 304 340 353 376 349 353 538 444 305 692 106 774 979,
101 560 557 675 965 682 301 030 257 770 383 367 178 916 685,
97 784 656 458 779 303 437 989 386 452 020 575 976 880 145,
128 032 200 604 782 244 713 434 477 043 518 271 901 000 964,
382 333 861 151 321 333 498 960 151 740 199 072 902 842 743,
1 761 584 675 590 839 734 287 976 390 632 540 296 552 968,
369 089 289 167 163 982 447 859 860 964 173 903 505 069 920};
t4 = Timing[rsaCracker[c4, n4, e4]]
```

Out[51]=
```
{1.14773, Null}
```

≈2^160-170

In[52]:=
```
n5 = 54 235 182 214 239 360 662 827 214 670 554 606 273 581 434 771 179;
e5 = 34 193;
c5 = {15 264 988 584 354 504 002 381 519 102 249 920 925 297 542 571 620,
   18 029 682 449 453 407 630 188 504 067 622 273 777 720 278 312 962,
   16 745 686 522 016 007 945 143 763 110 286 973 700 033 336 588 445,
   9 000 653 411 207 113 417 901 001 586 130 375 465 953 789 408 926,
   52 842 584 089 996 143 069 489 338 673 485 170 868 915 747 854 189,
   1 123 114 285 901 986 355 859 211 347 670 012 353 846 156 330 899,
   32 988 023 611 033 071 383 540 838 610 367 846 886 550 192 863 303,
   38 164 422 839 165 937 885 873 830 485 768 383 870 414 151 530 234,
   22 919 024 606 066 720 319 739 008 276 795 340 539 137 581 277 918,
   40 454 054 044 591 915 584 945 545 022 885 213 983 436 684 864 468,
   32 988 023 611 033 071 383 540 838 610 367 846 886 550 192 863 303,
   32 988 023 611 033 071 383 540 838 610 367 846 886 550 192 863 303,
   1 123 114 285 901 986 355 859 211 347 670 012 353 846 156 330 899,
   12 203 959 172 954 045 808 294 870 349 405 458 853 996 735 929 447,
   38 164 422 839 165 937 885 873 830 485 768 383 870 414 151 530 234,
   15 264 988 584 354 504 002 381 519 102 249 920 925 297 542 571 620,
   1 123 114 285 901 986 355 859 211 347 670 012 353 846 156 330 899,
   38 164 422 839 165 937 885 873 830 485 768 383 870 414 151 530 234,
   1 123 114 285 901 986 355 859 211 347 670 012 353 846 156 330 899,
   38 164 422 839 165 937 885 873 830 485 768 383 870 414 151 530 234,
   42 330 253 613 154 482 144 649 582 686 630 114 430 047 387 089 712,
   1 123 114 285 901 986 355 859 211 347 670 012 353 846 156 330 899,
   32 988 023 611 033 071 383 540 838 610 367 846 886 550 192 863 303,
   32 988 023 611 033 071 383 540 838 610 367 846 886 550 192 863 303,
   38 164 422 839 165 937 885 873 830 485 768 383 870 414 151 530 234,
   38 419 744 186 469 197 448 332 273 094 597 952 651 037 947 906 695,
   40 454 054 044 591 915 584 945 545 022 885 213 983 436 684 864 468,
   18 029 682 449 453 407 630 188 504 067 622 273 777 720 278 312 962,
   16 745 686 522 016 007 945 143 763 110 286 973 700 033 336 588 445,
   15 578 916 254 392 856 807 395 020 416 852 965 889 801 719 921 945};
t5 = Timing[rsaCracker[c5, n5, e5]]
```

Out[55]=
```
{7.46124, Null}
```

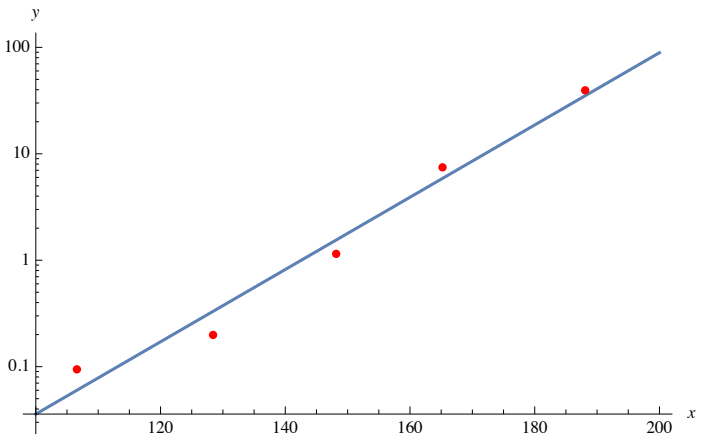
≈2^180-190

In[56]:=
```
n6 = 417 063 898 694 486 976 761 053 655 523 484 298 342 714 637 572 968 784 351;
e6 = 83 617;
c6 = {379 116 752 947 071 088 366 645 199 731 096 135 570 076 174 090 816 896 544,
   155 267 428 014 292 022 247 353 302 511 718 185 349 989 132 594 260 237 382,
   365 307 583 133 441 377 987 836 909 776 830 333 965 064 319 817 918 722 654,
   303 967 734 223 019 248 103 756 610 000 597 560 819 163 464 817 467 528 459,
   118 140 032 624 636 991 396 926 823 803 701 203 561 664 137 199 324 117 958,
   241 625 078 244 336 788 836 684 080 336 836 138 967 582 684 739 037 863 078,
   260 948 399 478 104 078 426 079 424 424 740 901 186 736 779 550 303 089 868,
   298 766 053 451 540 888 156 912 571 072 921 794 684 198 740 761 654 161 723,
   271 094 480 440 577 478 535 512 435 150 417 124 477 624 021 954 317 449 794,
   362 907 891 076 895 825 350 997 615 458 225 622 474 040 048 924 946 066 064,
   260 948 399 478 104 078 426 079 424 424 740 901 186 736 779 550 303 089 868,
   260 948 399 478 104 078 426 079 424 424 740 901 186 736 779 550 303 089 868,
   241 625 078 244 336 788 836 684 080 336 836 138 967 582 684 739 037 863 078,
   260 927 734 155 115 820 782 777 023 264 683 333 009 693 161 914 725 620 556,
   298 766 053 451 540 888 156 912 571 072 921 794 684 198 740 761 654 161 723,
   379 116 752 947 071 088 366 645 199 731 096 135 570 076 174 090 816 896 544,
   241 625 078 244 336 788 836 684 080 336 836 138 967 582 684 739 037 863 078,
   298 766 053 451 540 888 156 912 571 072 921 794 684 198 740 761 654 161 723,
   241 625 078 244 336 788 836 684 080 336 836 138 967 582 684 739 037 863 078,
   298 766 053 451 540 888 156 912 571 072 921 794 684 198 740 761 654 161 723,
   404 316 456 239 330 105 629 483 822 994 032 409 286 858 635 917 990 910 659,
   241 625 078 244 336 788 836 684 080 336 836 138 967 582 684 739 037 863 078,
   260 948 399 478 104 078 426 079 424 424 740 901 186 736 779 550 303 089 868,
   260 948 399 478 104 078 426 079 424 424 740 901 186 736 779 550 303 089 868,
   298 766 053 451 540 888 156 912 571 072 921 794 684 198 740 761 654 161 723,
   328 270 421 285 728 571 393 338 186 522 711 664 976 693 900 581 169 752 324,
   362 907 891 076 895 825 350 997 615 458 225 622 474 040 048 924 946 066 064,
   155 267 428 014 292 022 247 353 302 511 718 185 349 989 132 594 260 237 382,
   365 307 583 133 441 377 987 836 909 776 830 333 965 064 319 817 918 722 654,
   276 052 810 399 286 991 217 157 411 716 830 345 300 354 222 743 934 934 847};
t6 = Timing[rsaCracker[c6, n6, e6]]
```

Out[59]=
```
{39.4707, Null}
```

In[975]:=
```
x = N[Log2[{n2, n3, n4, n5, n6}]];
y = Part[{t2, t3, t4, t5, t6}, All, 1];
data = Transpose[{x, y}];
dataplot = ListLogPlot[data, AxesLabel → {x, y}, PlotStyle → Red];
data2 = Transpose[{x, Log[y]}];
solution = FindFit[data2, a x + b, {a, b}, x];
y[x_] = e^(a x+b) /. solution;
modelplot = LogPlot[y[x], {x, 100, 200}, AxesLabel → {x, y}];
Show[modelplot, dataplot]
y[1024]
y[2048]
y[4096]
```

Out[983]=



Out[984]= 8.46925×10^{29}

Out[985]= 4.97244×10^{64}

Out[986]= 1.71403×10^{134}