Many of the following exercises require a computer (PARI/GP or SAGEMATH).

3. Trial division factoring algorithm

1. Let n be a composite positive integer.
   (a) Verify that the complexity of the trial division algorithm, where we divide n by all primes $p \leq \sqrt{n}$ in increasing order until a factor is found (assuming that the list of such primes is available) is

$$\mathcal{O}(p \log n),$$

   where p is the smallest factor of n.
   (b) Verify that the complexity of the trial division algorithm, where we divide n by all positive integers $s \leq \sqrt{n}$ in increasing order until a factor is found, is

$$\mathcal{O}(p \log p \log n).$$

2. Let $n = p \cdot q$ be an integer, product of two primes $p \ll q$.
   (a) Estimate the variation in the amount of calculations to factor n with trial division, when the smallest factor p increases by one digit and q remains the same.
   (b) Estimate the variation in the amount of calculations to factor n with trial division, when the smallest factor p remains the same and q increases by one digit.
   (c) Do some experiments with PARI/GP (use *naif(n)* and time them).

3. On the base of the estimates of the previous exercise, exhibit integers whose factorisation with trial division requires one month, one year, 10 years.

4. Pollard $p - 1$

1. Let $K = 2^3 \cdot 3^2 \cdot 5 \cdot 7$.
   (a) Let $n = 95431706263$. Take a random $\overline{a} \in \mathbf{Z}_n^*$. Compute $\overline{b} = \overline{a}^K \bmod n$. Compute the divisor $d = \gcd(b-1, n)$ of n and the cofactor n/d.
   (b) Let $n = 57841557763361$. Take a random $\overline{a} \in \mathbf{Z}_n^*$. Compute $\overline{b} = \overline{a}^K \bmod n$. Compute the divisor $d = \gcd(b-1, n)$ of n and the cofactor n/d.
   (c) Why does the algorithm find these factorisations?

2. (use $pminus(n, B)$) Factor as much as possible the three following integers by the Pollard $p-1$ method, by progressively increasing the smoothness bound B.

$n_1 = 6480944046717780649546042565570850196336358017836292549973706514596045453 91$

$n_2 = 87008594415418296109798331073355399764294863864171215809269723035536733 8367$

$n_3 = 390802951911189150181349589384151083467496228819995634385579417637773837879970 06$
$8136035919305517302338111572218251 71$

$n_4 = 57910112155118020986691000616209882696955852203304284420354228211689903$
$410930476922872403336826857428628 01$

   (b) For a prime factor p found by this algorithm, compare the prime decomposition of $p-1$ with the smoothness bound B.
   (c) Verify that, in the successful cases, the algorithm breaks the number n as $m * q$ where m is the product of all prime factors p, all of which have the property that $p-1$ is B-smooth.

5. Pollard ρ

1. Let n be a composite integer of 200 digits. After 10000 iterations the Pollard ρ algorithm did not find any factors. What can we conclude about the size of the prime factors?
2. Let n be a composite integer of 200 digits, and let $B = 10000$. After how many unsuccesful iterations of the Pollard ρ algorithm we can conclude that probably n has no factor $\leq B$?
3. Let $n = p \cdot q$ be an integer, product of two primes $p \ll q$.
   (a) Estimate the variation in the amount of calculations to factor n with Pollard ρ, when the smallest factor p increases by one digit and q remains the same.
   (b) Estimate the variation in the amount of calculations to factor n with Pollard ρ, when the smallest factor p remains the same and q increases by one digit.
   (c) Do some experiments with PARI/GP (use *pollard2(n)*) and time them).
4. (a) (use *pollard2(n)*) Factor the number $n = 2107971466920603317676768413248655563326842644339$.
   (b) Run the algorithm several times: observe that the initial point changes, while the number of iterates which produce a given factor remains more or less of the same size.
5. Using the Pollard ρ algorithm,
   (a) factor the Mersenne numbers $M_n = 2^n - 1$ per $1 \leq n \leq 20$;
   (b) factor the Fermat numbers $F_n = 2^{2^n} + 1$, per $1 \leq n \leq 8$.
6. Factor completely the integers (combine the Pollard ρ algorithm with the Miller-Rabin test):

   $n_1 = 1077471755063687$

   $n_2 = 10687643000000395442791$

   $n_3 = 1000587688300000000000000000000000008574912503519308271870011$

   $n_4 = 10760671013000000000000000000000000000000000000000000000000000000000000000000000000000$
   000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   008242211154734950591104421572504 83.

   $n_5 = 10000076067099100000000000000000000000000000000000000000000000000000000000000000000000$
   000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   007659628141113788389728281238427868281.