Homework 7

Chris Powell

A. Write a program that takes as input a positive integer n and computes $\varphi(n)$. You may use brute force.

```
def gcd(a, b):
    """Return gcd(a,b)"""
    while b:
        a, b = b, a % b
    return a

def totient(n):
    """Given positive integer n, return totient(n)"""
    t = 0
    for i in range(1, n):
        if gcd(i, n) == 1:
            t += 1
    return t
```

Proof. Recall that termination and correctness of `gcd` has already been shown. Since `totient` iterates using `for`, it is clear that the algorithm must terminate. It remains to show that `totient` gives the correct output. Let t_k be the value of `t` after k iterations. If $\gcd(i, n) = 1$, then $t_{k+1} = t_k + 1$; otherwise, $t_{k+1} = t_k$. So

$$t_n = \sum_{\substack{1 \leqslant i \leqslant n \\ \gcd(i,n)=1}} i.$$

But this is the totient $\varphi(n)$, by definition. Therefore, since the algorithm returns t_n, it gives the correct output. $\square$

B. Compute

1. $\varphi(81)$

Observe that

$$\begin{aligned}
\varphi(81) &= \varphi\left(3^4\right) \\
&= 3^4 - 3^3 && \text{(Theorem 11.1.a)} \\
&= 81 - 27 \\
&= 54.
\end{aligned}$$

2. $\varphi(20736)$

Observe that

$$\begin{aligned}
\varphi(20736) &= \varphi\left(2^8 \cdot 3^4\right) \\
&= \varphi\left(2^8\right)\varphi\left(3^4\right) && \text{(Theorem 11.1.b)} \\
&= \left(2^8 - 2^7\right)\left(3^4 - 3^3\right) && \text{(Theorem 11.1.a)} \\
&= (256 - 128)(81 - 27) \\
&= (128)(54) \\
&= 6912.
\end{aligned}$$

3. $\varphi(10000000000)$

Observe that

$$\begin{aligned}
\varphi(1000000000000) &= \varphi\left(10^{12}\right) \\
&= \varphi\left((2 \cdot 5)^{12}\right) \\
&= (2 \cdot 5)^{12} \cdot \left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{5}\right) \\
&= (2 \cdot 5)^{12} \cdot \frac{1}{2} \cdot \frac{4}{5} \\
&= (2 \cdot 5)^{12} \cdot \frac{4}{10} \\
&= (2 \cdot 5)^{11} \cdot 4 \\
&= 10^{11} \cdot 4 \\
&= 40^{11}.
\end{aligned}$$

C. 1. Find all n for which $\varphi(n) = 4$.

Observe that

$$\begin{aligned}\varphi(n) &= \varphi\left(\prod_{i=1}^{r} p_i^{e_i}\right)\\ &= \prod_{i=1}^{r} \varphi\left(p_i^{e_i}\right)\\ &= \prod_{i=1}^{r} \left(p_i^{e_i} - p_i^{e_i-1}\right)\\ &= \prod_{i=1}^{r} \left((p-1)p_i^{e_i-1}\right)\end{aligned}$$

So $(p-1) \mid \varphi(n) = 4$. But

$$\{\, p \in \mathbb{N} \mid p \text{ is prime}, p-1 \text{ divides } 4 \,\} = \{\, 2, 3, 5 \,\}.$$

We consider the pairs of the powers of such of p:

$$\begin{aligned}\varphi(10) &= \varphi(2^1 \cdot 5^1) = \varphi(2^1)\varphi(5^1) = (2^1 - 2^0)(5^1 - 5^0) = 1 \cdot 4 = 4\\ \varphi(12) &= \varphi(2^2 \cdot 3^1) = \varphi(2^2)\varphi(3^1) = (2^2 - 2^1) \cdot (3^1 - 3^0) = 2 \cdot 2 = 4\\ \varphi(5) &= \varphi(2^0 \cdot 5^1) = 5^1 - 5^0 = 4.\\ \varphi(8) &= \varphi(2^3 \cdot 3^0) = \varphi(2^3)\varphi(3^0) = (2^3 - 2^2) = 8 - 4 = 4.\end{aligned}$$

We do not need to consider 3^e for $e \geqslant 2$ since $\varphi(3^2) = 3^2 - 3^1 = 6 > 4$. By similar reasoning, we do not need to consider 2^e for $e \geqslant 4$, nor 5^e for $e \geqslant 2$. Hence $\{\, n \in \mathbb{N} \mid \varphi(n) = 4 \,\} = \{\, 5, 8, 10, 12 \,\}$.

2. Find all n for which $\varphi(n) = 6$.

Observe that

$$\begin{aligned}
\varphi(n) &= \varphi\left(\prod_{i=1}^{r} p_i^{e_i}\right) \\
&= \prod_{i=1}^{r} \varphi\left(p_i^{e_i}\right) \\
&= \prod_{i=1}^{r} \left(p_i^{e_i} - p_i^{e_i-1}\right) \\
&= \prod_{i=1}^{r} \left((p-1)p_i^{e_i-1}\right)
\end{aligned}$$

So $(p-1) \mid \varphi(n) = 6$. But

$$\{\, p \in \mathbb{N} \mid p \text{ is prime}, p-1 \text{ divides } 6 \,\} = \{\, 2, 3, 7 \,\}.$$

We consider pairs of the powers of such of p:

$$\begin{aligned}
\varphi(7) &= \varphi(2^0 \cdot 7^1) = 7^1 - 7^0 = 6 \\
\varphi(9) &= \varphi(2^0 \cdot 3^2) = 3^2 - 3^1 = 9 - 3 = 6 \\
\varphi(14) &= \varphi(2^1 \cdot 7^1) = \varphi(2^1)\varphi(7^1) = (2^1 - 2^0)(7^1 - 7^0) = 1 \cdot 6 = 6 \\
\varphi(18) &= \varphi(2^1 \cdot 3^2) = \varphi(2^1)\varphi(3^2) = (2^1 - 2^0)(3^2 - 3^1) = 1 \cdot 6 = 6
\end{aligned}$$

We do not need to consider 3^e for $e \geqslant 3$, nor 7^e for $e \geqslant 2$ as their totient will be greater than 6. Hence $\{\, n \in \mathbb{N} \mid \varphi(n) = 6 \,\} = \{\, 7, 9, 14, 18 \,\}$.

D. Silverman 11.5 For each part, find an x that solves the given simultaneous congruences.

(a) $x \equiv 3 \pmod 7$ and $x \equiv 5 \pmod 9$

Since $\gcd(7, 9) = 1$, the Chinese Remainder Theorem implies there is a unique $x \in \mathbb{Z}/(7 * 9)\mathbb{Z}$ satisyfing the given system of congruences. To find x, we apply the `crt` algorithm developed in exercise E. We obtain $x = \texttt{crt(3,5,7,9)} = 59$.

(b) $x \equiv 3 \pmod{37}$ and $x \equiv 1 \pmod{87}$

> Since $\gcd(37, 87) = 1$, the Chinese Remainder Theorem implies there is a unique $x \in \mathbb{Z}/(37 * 87)\mathbb{Z}$ satisyfing the given system of congruences. To find x, we apply the `crt` algorithm developed in the following exercise E. We obtain $x =$ `crt(3,1,37,87)` $= 262$.

(c) $x \equiv 5 \pmod 7$ and $x \equiv 2 \pmod{12}$ and $x \equiv 8 \pmod{13}$

E. Silverman 11.8. You may not use brute force. Write a program that takes as input four integers (b, c, m, n) with $\gcd(m, n) = 1$ and computes an integer x with $0 \leqslant x \leqslant mn$ satisfying

$$x \equiv b \pmod m \quad \text{and} \quad x \equiv c \pmod n.$$

```python
def xgcd(a, b):
    """Return (g, x, y) such that a*x + b*y = g = gcd(a, b)"""
    if b == 0:
        return a, 1, 0
    x, g, v, w = 1, a, 0, b
    while w != 0:
        x, g, v, w = v, w, x - (g // w) * v, g % w
    x = x % (b // g)
    return g, x, (g - (a * x)) // b


def crt(a, b, m, n):
    """Given integers a,b,m,n, with gcd(m,n)=1, return unique
        x cong a,b (mod m,n)"""
    (g, r, s) = xgcd(m, n)
    return ((a * s * n) + (b * r * m)) % (m * n)
```

Proof. Recall that the termination and correctness of `xgcd` has already been shown. It immediately follows that `crt` must terminate. We show that `crt` gives the correct output. By the correctness of `xgcd`, r and s are such that

$$rm + sn = \mathrm{gcd}(m, n) = 1.$$

This implies $rm \equiv 1 \pmod{n}$ and $sn \equiv 1 \pmod{m}$. So $a(rm) \equiv a \pmod{m}$ and $b(rm) \equiv b \pmod{n}$. The algorithm returns $a(rn) + b(sm) \pmod{mn}$. But

$$a(sn) + b(rm) \equiv a(sn) \equiv a(1) \equiv a \pmod{m}$$

and

$$a(sn) + b(rm) \equiv b(rm) \equiv b(1) \equiv b \pmod{n}.$$

We've already shown that when $\gcd(m, n) = 1$, the map $[x]_{mn} \mapsto ([x]_m, [x]_n)$ is a bijection $\mathbb{Z}/mn \to \mathbb{Z}/m \times \mathbb{Z}/n$. Hence, the algorithm returns the correct output. □

F. Silverman 11.9 Let m_1, m_2, m_3 be positive integers such that each pair is relatively prime. That is,

$$\gcd(m_1, m_2) = 1 \quad \text{and} \quad \gcd(m_1, m_3) = 1 \quad \text{and} \quad \gcd(m_2, m_3) = 1.$$

Let a_1, a_2, a_3 be any three integers. Show that there is exactly one integer x in the interval $0 \leqslant x < m_1 m_2 m_3$ that simultaneously solves the three congruences

$$x \equiv a_1 \pmod{m_1}, \quad x \equiv a_2 \pmod{m_2}, \quad x \equiv a_3 \pmod{m_3}.$$

Can you figure out how to generalize this problem to deal with lots of congruences

$$x \equiv a_1 \pmod{m_1}, \quad x \equiv a_2 \pmod{m_2}, \ldots, \quad x \equiv a_r \pmod{m_r}?$$

In particular, what conditions do the moduli $m_1, m_2, \ldots, m_r$ need to satisfy?

G. Show that if $\gcd(m, n) > 1$, then

$$\begin{aligned} \psi : \mathbb{Z}/mn &\to \mathbb{Z}/m \times \mathbb{Z}/n \\ [x] &\mapsto ([x], [x]) \end{aligned}$$

is never bijective.

Proof. Assume $g = \gcd(m, n) > 1$. Then $g \mid mn$ since $g \mid m, n$. So $mn = gd$ for some $d \in \mathbb{Z}$. Thus $d = \frac{m}{g}n = m\frac{n}{g}$. We know $\frac{m}{g}, \frac{n}{g} \in \mathbb{Z}$ since $g \mid m, n$. Therefore $m, n \mid d$. So $d \equiv 0 \pmod{m}$ and $d \equiv 0 \pmod{n}$. So $\psi([d]_{mn}) = ([0]_m, [0]_n)$. But $1 < d < mn$ since $\gcd(m, n) > 1$. So $[d]_{mn} \neq [0]_{mn}$. Hence, ψ is not injective. $\square$