N1

$\forall M, N \quad \lambda x. MN =_\beta S(\lambda x. M)(\lambda x. N)$

$S(\lambda x. M)(\lambda x. N) = (\lambda fgx.\ fx(gx))(\lambda x. M)(\lambda x. N)$

$=_\beta \lambda x. (\lambda x. M)x((\lambda x. N)x) =_\beta \lambda x. MN$

N2 $xor = \lambda xy.\ x\ (not\ y)\ y$

Proof of correctness

(1) xor false false $=_\beta$ false (not false) false $=_\beta$ false

(2) xor false true $=_\beta$ false (not true) true $=_\beta$ true

(3) xor true false $=_\beta$ true (not false) false $=_\beta$ not false $=_\beta$ true

(4) xor true true $=_\beta$ true (not true) true $=_\beta$ not true $=_\beta$ false

It is commutative, because we considered all possible inputs and (1) = (1), (2) = (3), and (4) = (4)

N5

multiply $= \lambda m n f x.\ n\,(m f)\ x$

Why it works:

Each Church numeral is represented as $n = \lambda f x.\ \underbrace{f \ldots (f\ x) \ldots)}_{n \text{ times}}$, we can treat "f" here as a function which increments our numeral by "one". If we replace "f" with mf, then we increase our numeral by m. So,

$$\lambda m n f x.\ n(mf)\,x =_\beta \lambda f x.\ \underbrace{mf\,(mf \ldots (mf\,x))}_{n \text{ times}}.$$

therefore, it will result in the multiplication

1 is unit:

$\forall N$: multiply $1\ N =_\beta \lambda f x.\ 1\,(N f)\,x =$

$= \lambda f x.\ (\lambda f x.\ f x)\,(N f)\,x =_\beta \lambda f x.\ N f x = N$

multiply $N\ 1 =_\beta \lambda f x.\ N\,(1 f)\,x = \lambda f x.\ N((\lambda f x. f x) f) x$

$=_\beta \lambda f x.\ N\,(\lambda x.\ f x)\,x =_\beta \lambda f x.\ \underbrace{(\lambda x. f x(\ldots((\lambda x. f x) x)))}_{N \text{ times}} =$

$\lambda f x.\ \underbrace{f(f(f \ldots (f x)))}_{N \text{ times}} = N$

№3

$\text{pred} = \lambda nfx.\ n\,(\lambda gh.\ h\,(g\,f))\,(\lambda u.x)\ I$

Why it works:

The most interesting lambda here is $\lambda gh.\,h(gf)$

Suppose $n = \lambda fx.\,\underbrace{f(f\ldots(f}_{n\text{ times}}x))$, then, if we apply arguments $\lambda gh.\,h(gf)$ and $\lambda u.x$,

we get $\lambda fx.\,\underbrace{(\lambda gh.\,h(gf))(\ldots(\lambda gh.\,h(gf))}_{n\text{ times}}(\lambda u.x)) =_\beta$

$=_\beta \lambda fx.\,\underbrace{(\lambda gh.\,h(gf))(\ldots(\lambda gh.\,h(gf))}_{n-1\text{ times}}(\lambda u.x)))\ldots)$.

After a sequence of reductions, we get this: $\lambda fx.\,(\lambda h.\,h\,\underbrace{f(f(\ldots(f}_{n-1\text{ times}}x))\ldots)$. So our result after reduction is lambda with body which consist of lambda. In order, to get regular Church numeral we can pass to h parameter identity combinator and thus retrieve a Church numeral eg.

$\lambda fx.\,(\lambda h.\,h\,\underbrace{f(f\ldots(fx))}_{n-1\text{ times}})\ \lambda x.x =_\beta \lambda fx.\,\underbrace{f(\ldots(fx))}_{n-1\text{ times}} = n-1$

The key idea here is that parameter "h" responsible for the inc function and when we are at the outermost expression, we can decide what to do with it

$$pred \circ succ \stackrel{?}{=} id \quad \forall n$$

$$pred = \lambda nfx.\, n\,(\lambda gh.\, h(gf))\,(\lambda u.x)\, I$$

$$succ = \lambda nfx.\, f(nfx)$$

$\forall N:$

$$pred\,(succ\ N) = pred\,((\lambda nfx.\, f(nfx))\ N)$$

$$\underset{\beta}{=} pred\,(\lambda fx.\, \underbrace{f(\ldots(fx))}_{n+1 \text{ times}}) \underset{def}{=} pred\,(N+1)$$

$$(\lambda nfx.\, n\,(\lambda gh.\, h(gf))\,(\lambda u.x)\, I)\,(N+1) \underset{\beta}{=}$$

$$\lambda fx.\, (N+1)\,(\lambda gh.\, h(gf))\,(\lambda u.x)\, I \underset{\beta}{=}$$

$$\lambda fx.\, [\underbrace{((\lambda gh.\, h(gf)) \ldots ((\lambda gh.\, h(gf))}_{N+1 \text{ times}}\, \lambda u.x)) \ldots]\ I \underset{\beta}{=}$$

$$\lambda fx.[\lambda h.\underbrace{f(f\ldots(f}_{N\text{ times}}\,x))\ldots]\,I \underset{\beta}{=} \lambda fx.\underbrace{f(f\ldots(f}_{N\text{ times}}\,x))\ldots)$$

$$= N = I\,N$$