Q# 0.10 Language Quick Reference

Primitive Types

64-bit integers	`Int`
Double-precision floats	`Double`
Booleans	`Bool` e.g.: `true` or `false`
Qubits	`Qubit`
Pauli basis	`Pauli` e.g.: `PauliI`, `PauliX`, `PauliY`, or `PauliZ`
Measurement results	`Result` e.g.: `Zero` or `One`
Sequences of integers	`Range` e.g.: `1..10` or `5..-1..0`
Strings	`String` e.g.: `"Hello Quantum!"`
"Return no information" type	`Unit` e.g.: `()`

Derived Types

Arrays	*elementType*`[]`
Tuples	`(`*type0*`,` *type1*`, ...)` e.g.: `(Int, Qubit)`
Functions	*input* `->` *output* e.g.: `ArcCos : (Double) -> Double`
Operations	*input* `=>` *output* `is` *variants* e.g.: `H : (Qubit => Unit is Adj)`

User-Defined Types

Declare UDT with anonymous items	`newtype` *Name* `= (Type, Type);` e.g.: `newtype` *Pair* `= (Int, Int);`
Define UDT literal	*Name*`(`*baseTupleLiteral*`)` e.g.: `let origin = Pair(0, 0);`
Unwrap operator ! (convert UDT to underlying type)	*VarName*`!` e.g.: `let originTuple = origin!;` (now `originTuple = (0, 0)`)

Declare UDT with named items	`newtype` *Name* `=`
	`(`*Name1*`: Type,` *Name2*`: Type);`
	e.g.: `newtype Complex =`
	`(Re : Double, Im : Double);`

Accessing named items of UDTs	*VarName*`::`*ItemName* e.g.: `complexVariable::Re`

Update-and-reassign for named UDT items	`set` *VarName* `w/=` *ItemName* `<-` *val*`;`
	e.g.: `mutable p = Complex(0., 0.);`
	`set p w/= Re <- 1.0;`

Symbols and Variables

Declare immutable symbol	`let` *varName* `=` *value*
Declare mutable symbol (variable)	`mutable` *varName* `=` *initialValue*
Update mutable symbol (variable)	`set` *varName* `=` *newValue*
Apply-and-reassign	`set` *varName* *operator*`=` *expression* e.g.: `set counter += 1;`

Functions and Operations

Define function (classical routine)	`function` *Name*`(`*in0* `:` *type0*`, ...)`
	`:` *returnType* `{`
	`//` *function body*
	`}`

Call function	*Name*`(`*parameters*`)` e.g.: `let two = Sqrt(4.0);`

Define operation (quantum routine) with explicitly specified body, controlled and adjoint variants	`operation` *Name*`(`*in0* `:` *type0*`, ...)`
	`:` *returnType* `{`
	`body { ... }`
	`adjoint { ... }`
	`controlled { ... }`
	`adjoint controlled { ... }`
	`}`

Define operation with automatically generated adjoint and controlled variants	`operation` *Name*`(`*in0* `:` *type0*`, ...)`
	`:` *returnType* `is Adj + Ctl {`
	`...`
	`}`

Call operation	*Name*`(`*parameters*`)` e.g.: `Ry(0.5 * PI(), q);`
Call adjoint operation	`Adjoint` *Name*`(`*parameters*`)` e.g.: `Adjoint Ry(0.5 * PI(), q);`

Call controlled operation	`Controlled` *Name*`(`*controlQubits*`,`
	parameters`)`
	e.g.: `Controlled Ry(controls,`
	`(0.5 * PI(), target));`

Control Flow

Iterate over a range of numbers	`for (`*index* `in` *range*`) {`
	`// Use integer` *index*
	`...`
	`}`
	e.g.: `for (i in 0..N-1) { ... }`

While loop (within functions)	`while (`*condition*`) {`
	`...`
	`}`

Iterate over an array	`for (`*val* `in` *array*`) {`
	`// Use value` *val*
	`...`
	`}`
	e.g.: `for (q in register) { ... }`

Repeat-until-success loop	`repeat { ... }`
	`until (`*condition*`)`
	`fixup { ... }`

Conditional statement	`if (`*cond1*`) { ... }`
	`elif (`*cond2*`) { ... }`
	`else { ... }`

Ternary operator	*condition* `?` *caseTrue* `|` *caseFalse*
Return a value	`return` *value*
Stop with an error	`fail "`*Error message*`"`
Conjugations ($ABA^\dagger$ pattern)	`within { ... }` `apply { ... }`

Arrays

Allocate array	`mutable` *name* `= new` *Type*`[`*length*`]` e.g.: `mutable b = new Bool[2];`
Get array length	`Length(`*name*`)`
Access k-th element	*name*`[k]` NB: indices are 0-based
Assign k-th element (copy-and-update)	`set` *name* `w/=` *k* `<-` *value* e.g.: `set b w/= 0 <- true;`
Array literal	`[`*value0*`,` *value1*`, ...]` e.g.: `let b = [true, false, true];`
Array concatenation	*array1* `+` *array2* e.g.: `let t = [1, 2, 3] + [4, 5];`

Slicing (subarray)	*name*`[`*sliceRange*`]`
	e.g.: if `t = [1, 2, 3, 4, 5]`, then
	`t[1 .. 3]` is `[2, 3, 4]`
	`t[3 ...]` is `[4, 5]`
	`t[... 1]` is `[1, 2]`
	`t[0 .. 2 ...]` is `[1, 3, 5]`
	`t[...-1...]` is `[5, 4, 3, 2, 1]`

Debugging (classical)

Print a string	`Message("Hello Quantum!")`
Print an interpolated string	`Message($"Value = {`*val*`}")`

Resources

Documentation

Quantum Development Kit	`https://docs.microsoft.com/quantum`
Q# Language Reference	`https://docs.microsoft.com/quantum/language`
Q# Libraries Reference	`https://docs.microsoft.com/qsharp/api`

Q# Code Repositories

QDK Samples	`https://github.com/microsoft/quantum`
QDK Libraries	`https://github.com/microsoft/QuantumLibraries`
Quantum Katas (tutorials)	`https://github.com/microsoft/QuantumKatas`
Q# compiler and extensions	`https://github.com/microsoft/qsharp-compiler`
Simulation framework	`https://github.com/microsoft/qsharp-runtime`
Jupyter kernel and Python host	`https://github.com/microsoft/iqsharp`
Source code for the documentation	`https://github.com/MicrosoftDocs/quantum-docs-pr`

Qubit Allocation

Allocate a register of N qubits	`using (reg = Qubit[N]) {`
	`// Qubits in reg start in |0⟩.`
	`...`
	`// Qubits must be returned to |0⟩.`
	`}`

Allocate one qubit	`using (one = Qubit()) { ... }`

Allocate a mix of qubit registers and individual qubits	`using ((x, y, ... ) =`
	`(Qubit[N], Qubit(), ... )) {`
	`...`
	`}`

Debugging (quantum)

Print amplitudes of wave function	`DumpMachine("dump.txt")`
Assert that a qubit is in $\vert 0\rangle$ or $\vert 1\rangle$ state	`AssertQubit(Zero, zeroQubit)` `AssertQubit(One, oneQubit)`

Measurements

Measure qubit in Pauli Z basis	`M(oneQubit)` yields a `Result` (`Zero` or `One`)
Reset qubit to $\vert 0\rangle$	`Reset(oneQubit)`
Reset an array of qubits to $\vert 0..0\rangle$	`ResetAll(register)`

Working with Q# from command line

Command Line Basics

Change directory	`cd dirname`
Go to home	`cd ~`
Go up one directory	`cd ..`
Make new directory	`mkdir dirname`
Open current directory in VS Code	`code .`

Working with Q# Projects

Create new project	`dotnet new console -lang Q# --output project-dir`
Change directory to project directory	`cd project-dir`
Build project	`dotnet build`
Run all unit tests	`dotnet test`

Math reference

Complex Arithmetic

i^2	-1
$(a+bi)+(c+di)$	$(a+c)+(b+d)i$
$(a+bi)(c+di)$	$a\cdot b+a\cdot di+c\cdot bi+(b\cdot d)i^2 =$ $a\cdot b-b\cdot d+(a\cdot d+c\cdot b)i$
Complex conjugate $x=a+bi$	$\overline{x}=a-bi$
Complex division $\frac{a+bi}{c+di}$	$\frac{x}{y}=\frac{x}{y}\cdot 1=\frac{x}{y}\cdot\frac{\overline{y}}{\overline{y}}=$ $=\frac{(a+bi)(c-di)}{(c+di)(c-di)}=\frac{(a+bi)(c+di)}{c^2+d^2}$
Modulus $x=a+bi$	$\vert x\vert=\sqrt{a^2+b^2}$
$e^{i\theta}$	$\cos(\theta)+i\sin(\theta)$
e^{a+bi}	$e^a\cdot e^{bi}=e^a(\cos(b)+i\sin(b))$
r^{a+bi}	$c^a\cdot c^{bi}=c^a\cdot e^{bi\ln c}=$ $=c^a(\cos(b\ln c)+i\sin(b\ln c))$
Polar form $re^{i\theta}$	Cartesian form $r(\cos(\theta)+i\sin(\theta))$ $a+bi$
Cartesian form $a+bi$	Polar form $re^{i\theta}$ $r=\sqrt{a^2+b^2}$ $\theta=\arctan(\frac{b}{a})$

Linear Algebra

N X M Matrix	$\begin{array}{cc} & \xrightarrow{\text{n columns}} \\ \text{m rows}\downarrow & \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}\end{array}$
Vector of size N	$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}$
Addition $\begin{bmatrix} a & b \\ c & d \end{bmatrix}+\begin{bmatrix} e & f \\ g & h \end{bmatrix}$	$\begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$
Scalar multiplication $a\cdot\begin{bmatrix} b & c \\ d & e \end{bmatrix}$	$\begin{bmatrix} a\cdot b & a\cdot c \\ a\cdot d & a\cdot e \end{bmatrix}$
Matrix multiplication $\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}\cdot\begin{bmatrix} x \\ y \\ z \end{bmatrix}$	$\begin{bmatrix} a\cdot x+b\cdot y+c\cdot z \\ d\cdot x+e\cdot y+f\cdot z \end{bmatrix}$
Transpose $\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}^T$	$\begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix}$
Adjoint $\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}^\dagger$	$\begin{bmatrix} \overline{a} & \overline{e} \\ \overline{b} & \overline{d} \\ \overline{c} & \overline{f} \end{bmatrix}$
Inner product $\left\langle \begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} c \\ d \end{bmatrix} \right\rangle$	$\begin{bmatrix} a \\ b \end{bmatrix}^\dagger \begin{bmatrix} c \\ d \end{bmatrix}=\begin{bmatrix} \overline{a} & \overline{b} \end{bmatrix}\begin{bmatrix} c \\ d \end{bmatrix}=$ $\overline{a}\cdot c+\overline{b}\cdot d$
Outer product $\begin{bmatrix} a \\ b \end{bmatrix}$ and $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$	$\begin{bmatrix} a \\ b \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix}^\dagger=\begin{bmatrix} a \\ b \end{bmatrix}\begin{bmatrix} \overline{x} & \overline{y} & \overline{z} \end{bmatrix}=$ $\begin{bmatrix} a\cdot\overline{x} & a\cdot\overline{y} & a\cdot\overline{z} \\ b\cdot\overline{x} & b\cdot\overline{y} & b\cdot\overline{z} \end{bmatrix}$

Gates reference

Single Qubit gates						
Gate	Matrix representation	Ket-bra representation	Applying to $\|\psi\rangle = \alpha\|0\rangle + \beta\|1\rangle$	Applying to basis states:	$\|0\rangle$, $\|1\rangle$, $\|+\rangle$, $\|-\rangle$ and	$\|\pm i\rangle = \frac{1}{\sqrt{2}}(\|0\rangle \pm i\|1\rangle)$
X	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$	$\|0\rangle\langle 1\| + \|1\rangle\langle 0\|$	$\|\psi\rangle = \alpha\|1\rangle + \beta\|0\rangle$	$X\|0\rangle = \|1\rangle$ $X\|1\rangle = \|0\rangle$	$X\|+\rangle = \|+\rangle$ $X\|-\rangle = -\|-\rangle$	$X\|i\rangle = i\|-i\rangle$ $X\|-i\rangle = -i\|i\rangle$
Y	$\begin{bmatrix}0 & -i\\i & 0\end{bmatrix}$	$i(\|1\rangle\langle 0\| - \|0\rangle\langle 1\|)$	$Y\|\psi\rangle = i(\alpha\|1\rangle - \beta\|0\rangle)$	$Y\|0\rangle = i\|1\rangle$ $Y\|1\rangle = -i\|0\rangle$	$Y\|+\rangle = -i\|-\rangle$ $Y\|-\rangle = i\|+\rangle$	$Y\|i\rangle = \|i\rangle$ $Y\|-i\rangle = -\|-i\rangle$
Z	$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$	$\|0\rangle\langle 0\| - \|1\rangle\langle 1\|$	$Z\|\psi\rangle = \alpha\|0\rangle - \beta\|1\rangle$	$Z\|0\rangle = \|0\rangle$ $Z\|1\rangle = -\|1\rangle$	$Z\|+\rangle = \|-\rangle$ $Z\|-\rangle = \|+\rangle$	$Z\|i\rangle = \|-i\rangle$ $Z\|-i\rangle = \|i\rangle$
I	$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$	$\|0\rangle\langle 0\| + \|1\rangle\langle 1\|$	$I\|\psi\rangle = \|\psi\rangle$			
H	$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$	$\|0\rangle\langle +\| + \|1\rangle\langle -\|$	$H\|\psi\rangle = \alpha\|+\rangle + \beta\|-\rangle =$ $\frac{\alpha+\beta}{\sqrt{2}}\|0\rangle + \frac{\alpha-\beta}{\sqrt{2}}\|1\rangle$	$H\|0\rangle = \|+\rangle$ $H\|1\rangle = \|-\rangle$	$H\|+\rangle = \|0\rangle$ $H\|-\rangle = \|1\rangle$	$H\|i\rangle = e^{i\pi/4}\|-i\rangle$ $H\|-i\rangle = e^{-i\pi/4}\|i\rangle$
S	$\begin{bmatrix}1 & 0\\0 & i\end{bmatrix}$	$\|0\rangle\langle 0\| + i\|1\rangle\langle 1\|$	$S\|\psi\rangle = \alpha\|0\rangle + i\beta\|1\rangle$	$S\|0\rangle = \|0\rangle$ $S\|1\rangle = i\|1\rangle$	$S\|+\rangle = \|i\rangle$ $S\|-\rangle = \|-i\rangle$	$S\|i\rangle = \|-\rangle$ $S\|-i\rangle = \|+\rangle$
T	$\begin{bmatrix}1 & 0\\0 & e^{i\pi/4}\end{bmatrix}$	$\|0\rangle\langle 0\| + e^{i\pi/4}\|1\rangle\langle 1\|$	$T\|\psi\rangle = \alpha\|0\rangle + e^{i\pi/4}\beta\|1\rangle$	$T\|0\rangle = \|0\rangle$ $T\|1\rangle = e^{i\pi/4}\|1\rangle$		
$R_x(\theta)$	$\begin{bmatrix}\cos\frac{\theta}{2} & -i\sin\frac{\theta}{2}\\-i\sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{bmatrix}$	$\cos\frac{\theta}{2}\|0\rangle\langle 0\| - i\sin\frac{\theta}{2}\|1\rangle\langle 0\|$ $- i\sin\frac{\theta}{2}\|0\rangle\langle 1\| + \cos\frac{\theta}{2}\|1\rangle\langle 1\|$	$R_x(\theta)\|\psi\rangle = (\alpha\cos\frac{\theta}{2} - i\beta\sin\frac{\theta}{2})\|0\rangle +$ $+ (\beta\cos\frac{\theta}{2} - i\alpha\sin\frac{\theta}{2})\|1\rangle$	$R_x(\theta)\|0\rangle =$ $\cos\frac{\theta}{2}\|0\rangle - i\sin\frac{\theta}{2}\|1\rangle$	$R_x(\theta)\|1\rangle =$ $\cos\frac{\theta}{2}\|1\rangle - i\sin\frac{\theta}{2}\|0\rangle$	
$R_y(\theta)$	$\begin{bmatrix}\cos\frac{\theta}{2} & -\sin\frac{\theta}{2}\\\sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{bmatrix}$	$\cos\frac{\theta}{2}\|0\rangle\langle 0\| + \sin\frac{\theta}{2}\|1\rangle\langle 0\|$ $- \sin\frac{\theta}{2}\|0\rangle\langle 1\| + \cos\frac{\theta}{2}\|1\rangle\langle 1\|$	$R_y(\theta)\|\psi\rangle = (\alpha\cos\frac{\theta}{2} - \beta\sin\frac{\theta}{2})\|0\rangle +$ $+ (\beta\cos\frac{\theta}{2} + \alpha\sin\frac{\theta}{2})\|1\rangle$	$R_y(\theta)\|0\rangle =$ $\cos\frac{\theta}{2}\|0\rangle + \sin\frac{\theta}{2}\|1\rangle$	$R_y(\theta)\|1\rangle =$ $\cos\frac{\theta}{2}\|1\rangle - \sin\frac{\theta}{2}\|0\rangle$	
$R_z(\theta)$	$\begin{bmatrix}e^{-i\theta/2} & 0\\0 & e^{i\theta/2}\end{bmatrix}$	$e^{-i\theta/2}\|0\rangle\langle 0\| + e^{i\theta/2}\|1\rangle\langle 1\|$	$R_z(\theta)\|\psi\rangle =$ $\alpha e^{-i\theta/2}\|0\rangle + \beta e^{i\theta/2}\|1\rangle$	$R_z(\theta)\|0\rangle = e^{-i\theta/2}\|0\rangle$	$R_z(\theta)\|1\rangle = e^{i\theta/2}\|1\rangle$	
$R_1(\theta)$	$\begin{bmatrix}1 & 0\\0 & e^{i\theta}\end{bmatrix}$	$\|0\rangle\langle 0\| + e^{i\theta}\|1\rangle\langle 1\|$	$R_1(\theta)\|\psi\rangle = \alpha\|0\rangle + \beta e^{i\theta}\|1\rangle$	$R_1(\theta)\|0\rangle = \|0\rangle$	$R_1(\theta)\|1\rangle = e^{i\theta}\|1\rangle$	