Numerical Methods 35006
Computer Lab 7: Multi-dimensional Integration

The first two questions in this Lab will rely on the `quadz` function that you programmed last week. If you are not confident in your implementation then transfer it over from last week's lab solutions, which are online.

1. Create code that uses nested Gaussian quadrature to integrate a 2D function

$$\int_a^b \int_c^d f(x,y)\mathrm{d}x\mathrm{d}y$$

using the function call

```
result = squad2d(f,a,b,c,d)
```

where `f(x,y)` is a two-variable function.

For this you should create two functions, one inside the other:

a) On the "inside", create a subfunction `yint(f,x,c,d)` that calls `quadz` to integrate the function `f(x,y)` over the variable `y` in the interval `[c,d]`.

c) On the "outside", create the main integration function `squad2d`, which calls `quadz` to integrate the function `yint` over the variable `x` in the interval `[a,b]`.

Test your 2d integrator by evaluating

$$\int_0^2 \int_0^1 x^2 y\mathrm{d}x\mathrm{d}y \ .$$

2. Think of another 2D function to test your `squadz` code on, then save it to your `myquad` module.

3. You can then generate a numpy array of 10 random numbers from a uniform random distribution between 0 and 1 with the function call

```
p = np.random.uniform(size=10)
```

By generating 10000 random numbers, create code that uses Monte-Carlo integration to perform the integral

$$\int_0^\pi \sin^2 x\mathrm{d}x$$

4. Create a function with the call

```
montyn(f,a,b,N)
```

that does a Monte-Carlo integration on a function tt f(x) over the interval `[a,b]` using `N` points.

5. Plot the value of the integral in the previous question for values of N up to 10000.

6. A sequence of random numbers in 2D can be computed using a call

```
p = np.random.uniform(size=[N,2])
```

a) Plot a set of 1000 points in 2D, uniformly distributed on the rectangular region given by $x \in [-1,1]$,$y \in [-1,1]$.

b) Create a function called `region(x,y)`, which returns `true` if a point `(x,y)` lies in the unit circle centred at the origin.

c) Plot the subset of random points from part (a) that lie in this region.

7. A quasi-random number set, using Sobol sequences, can be generated using the `qmc` package, which is imported using

```
from scipy.stats import qmc
```

A sequence of quasi-random numbers in 2D, lying between 0 and 1, can then be computed using the call

```
M = 10 sampler = qmc.Sobol(d=2) # set
up the sequence
p = sampler.random_base2(10) #generate
2**M points
```

Modify your code in Q6 to generate a set of 1024 quasi-random numbers, and plot the result.

8. Using your code in Q7 to create a 2D Monte-Carlo integrator, which can be called with the function call

```
result = monty2d(f,region,a,b,M)
```

for 2^M points, and where you give the region for Monte-Carlo sampling as being in the rectangle with edges at $[a,b]$. Test this code by computing the area of a unit circle. Save your `monty2d` function to the `myquad` module.