

## Physics 350 Lab 10 Sample Solution

```
IDL> ; PROBLEM 1, PART A
IDL> A = [ [0, 1, 1], $
IDL> [2, 3, 2], $
IDL> [4, 5, -1] ]
IDL> print,a
      0      1      1
      2      3      2
      4      5     -1
IDL> print, a[1,2]
      5

IDL> ; PROBLEM 1, PART B
IDL> ;
IDL> B = [ [2, -2, 1], [1, 0, 1], [-1, 1, 0] ]
IDL> print, b
      2     -2      1
      1      0      1
     -1      1      0
IDL> print, A ## B
      0      1      1
      5     -2      5
     14     -9      9

IDL> ; PROBLEM 1, PART C
IDL> ;
IDL> print, A*B
      0     -2      1
      2      0      2
     -4      5      0
IDL> ;
IDL> ; To get A*B, IDL multiplies A and B element by element. So the 1st row,
IDL> ; 1st column of the result is 1st row, 1st column of A times
IDL> ; 1st row, 1st column of B and so on.

IDL> ; PROBLEM 2
IDL> ;
IDL> ; First define a function RotationZ that returns a matrix for a
IDL> ; rotation about the z axis by angle theta. The code for this is attached
IDL> ; in the file rotationz.pro
IDL> ;
IDL> ; based on Part 2 of lab 8 we define a matrix M1 given by
IDL> ;
IDL> M1 = RotationZ(!pi/4)
Compiled module: ROTATIONZ.
IDL> test = [ 1, 1, 1]
IDL> print, M1 ## test
      0.000000
      1.41421
      1.00000

IDL> ;
IDL> ; Following along in Lab 8, we now define a matrix M2
IDL> ;
IDL> M2 = [ [1, 0, 0], [0, 1/sqrt(3), -sqrt(2/3)], [0, sqrt(2/3), 1/sqrt(3)] ]
IDL> print, m2
      1.00000      0.000000      0.000000
      0.000000      0.577350      0.000000
      0.000000      0.000000      0.577350
IDL> ;
```

## Physics 350 Lab 10 Sample Solution

```
IDL> ; Hmmm, that sure is funny. What happened to my off-diagonal elements?
IDL> ; AHA! I wrote sqrt(2/3), and 2/3 is evaluated as an integer, so 2/3=0 and
IDL> ; the square root of 0 is 0. Better fix that up...
IDL> ;
IDL> M2 = [ [1, 0, 0], [0, 1/sqrt(3), -sqrt(2.0/3)], [0, sqrt(2.0/3), 1/sqrt(3)] ]
IDL> print, m2
      1.00000      0.000000      0.000000
      0.000000      0.577350     -0.816497
      0.000000      0.816497      0.577350
IDL> ;
IDL> ; Much better now. Next, according to lab 8, we define M to be the product of
IDL> ; M1 and M2.
IDL> ;
IDL> M = M2 ## M1
IDL> print, M
      0.707107     -0.707107      0.000000
      0.408248      0.408248     -0.816497
      0.577350      0.577350      0.577350
IDL> ; Make sure M really does what we want...
IDL> ;
IDL> print, M ## test
      0.000000
     -5.96046e-008
      1.73205
IDL> ;
IDL> ; We should get [0, 0, sqrt(3)]. Note that the second element in our
IDL> ; result is roughly 6E-8 (which means 6 times 10 to the -8).
IDL> ; This is as close
IDL> ; as the computer can get to zero when using 4 byte numbers
IDL> ; to do the math.
IDL> ; MAPLE hid this sort of thing from you by using higher precision to
IDL> ; do its arithmetic.
IDL> ;
IDL> ;
IDL> ; We also need the inverse of M,
IDL> ;
IDL> Minv = invert(M)
IDL> print, Minv
      0.707107      0.408248      0.577350
     -0.707107      0.408248      0.577350
      4.21468e-008     -0.816496      0.577350
IDL> ;
IDL> ; The overall rotation matrix is Minv R M,
IDL> ; and since we would like to know the value of S for several different angles phi
IDL> ; (see lab 8 solutions for definition of phi), I'll write a small function
IDL> ; Srot that takes in an angle and gives back the matrix S.
IDL> ; I will need to include M as part of the, or pass it in to the function as an
IDL> ; argument. In the attached srot.pro I went for the second option, making M and
IDL> ; argument to srot. Choosing the other route is fine too.
IDL> ;
IDL> ; See the file srot.pro below for details.
IDL> ;
IDL> ; Now some examples...first try phi=120 degrees. Remember we need to write
IDL> ; the angle in radians.
IDL> S120 = Srot( 120*!pi/180, M )
IDL> print, S120
     -3.28060e-008  9.01970e-009      1.00000
      1.00000     -6.13677e-008  1.24403e-008
```

## Physics 350 Lab 10 Sample Solution

```
2.29260e-008      1.00000 1.24403e-008
IDL> ;
IDL> ; Notice most of the entries are very close, but not quite, zero.
IDL> ; Let me clean this up by picking out the small elements and setting them to
IDL> ; zero...this goes beyond what you were expected to do.
IDL> ;
IDL> smalls = where( abs(s120) lt 1e-6 )
IDL> s120[smalls]=0
IDL> print,s120
      0.000000      0.000000      1.000000
      1.000000      0.000000      0.000000
      0.000000      1.000000      0.000000
IDL> ; Looks much better now!
IDL> ;
IDL> ;
IDL> ; Next we see where [1, 2, 4] goes after rotation by 60 degrees...
IDL> S60 = Srot( 60*!pi/180, M )
IDL> vec124 = [1., 2, 4]
IDL> ; Notice I wrote 1. not 1 to make sure vec124 is floating point.
IDL> print, S60 ## vec124
      2.66667
      0.666666
      3.66667
IDL> ; Excellent! It matches MAPLE.
IDL> ;
IDL> ; Finally, we see where [1, 1, 0] goes after a 90 degree rotation.
IDL> ;
IDL> S90 = Srot( !pi/2, M )
IDL> vec110 = [1., 1, 0]
IDL> vec110_new = S90 ## vec110
IDL> print, vec110_new
      0.0893164
      1.24402
      0.666667
IDL> ;
IDL> ; It isn't obvious to me that this is the same answer we got in MAPLE,
IDL> ; so I will put in the vector vec110_new from MAPLE to make
IDL> ; sure we are really getting the same result.
IDL> vec110_new_maple = [ 2./3-1/sqrt(3), 2./3+1/sqrt(3), 2./3]
IDL> print, vec110_new_maple
      0.0893164      1.24402      0.666667
IDL> ; This matches out IDL result perfectly!
```

## Physics 350 Lab 10 Sample Solution

---

contents of file rotationz.pro:

```
FUNCTION RotationZ, theta
; the angle theta should be in radians

; Note the liberal use of continuation symbols $ to spread the code
; over multiple lines so it is easier to read.
  Rz = [ $
        [ cos(theta),-sin(theta), 0 ], $
        [ sin(theta), cos(theta), 0 ], $
        [ 0, 0, 1 ] $
      ]

  RETURN, Rz
END
```

---

contents of file srot.pro:

```
FUNCTION srot, phi, M
;
; S = Minv R M,
; where R is a rotation by angle phi about the z axis
; and Minv is the inverse of M.
;
  R = RotationZ(phi)
  Minv = invert(M)

  RETURN, Minv ## R ## M
END
```