

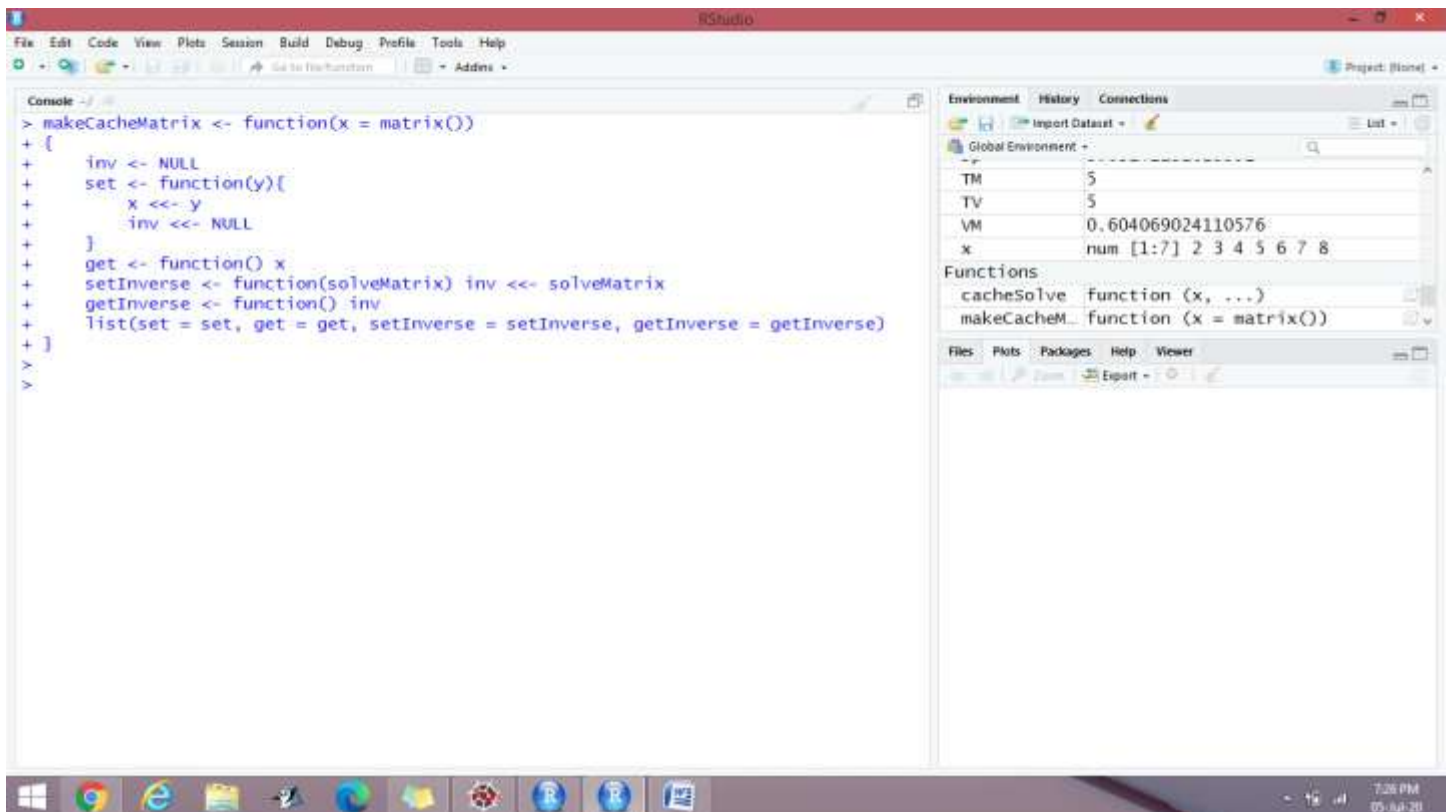
## Assignment: Caching the Inverse of a Matrix

Following are the 2 topics of the assignment to write a pair of functions that cache the inverse of the matrix

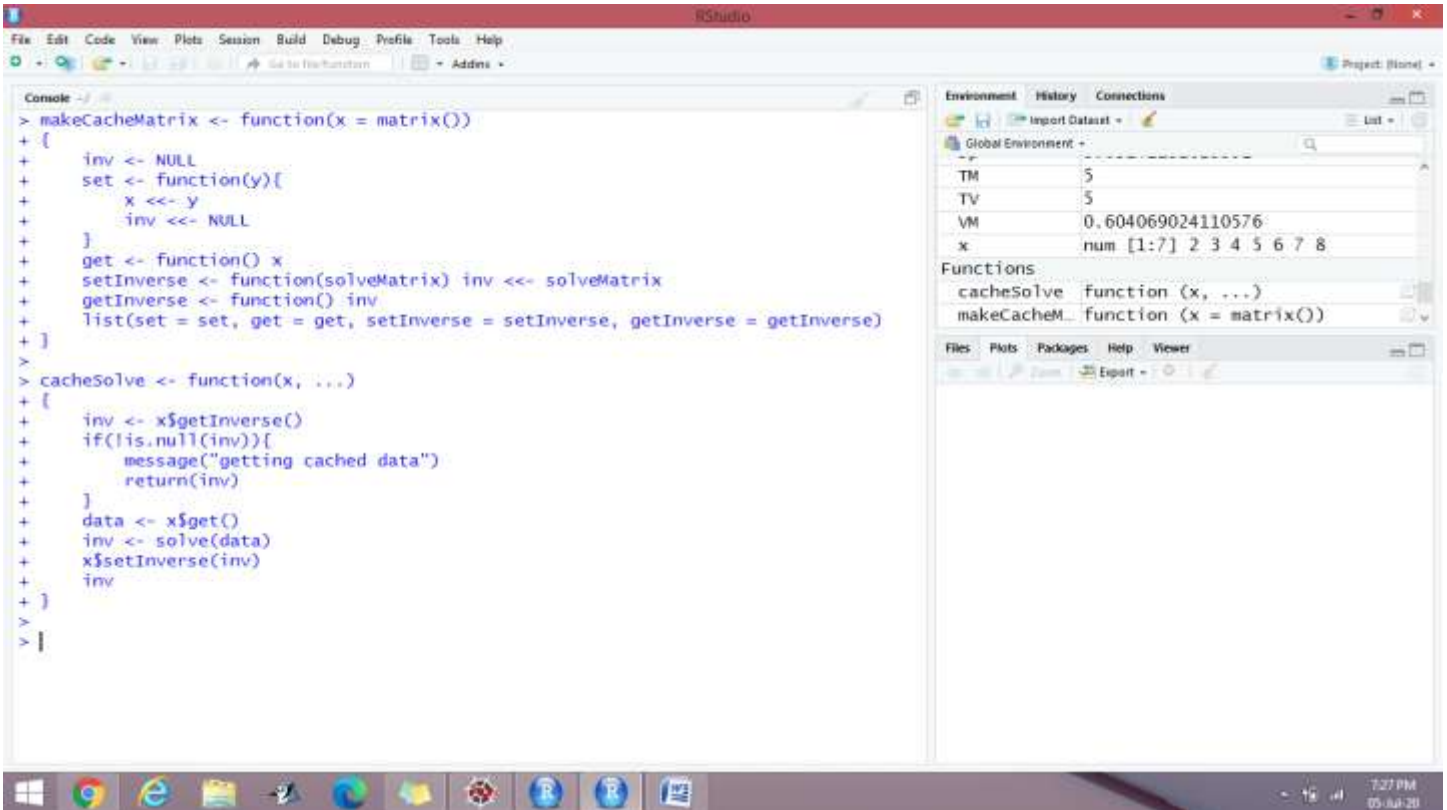
Write the following functions:

1. `makeCacheMatrix`: This function creates a special "matrix" object that can cache its inverse.
2. `cacheSolve`: This function computes the inverse of the special "matrix" returned by `makeCacheMatrix` above. If the inverse has already been calculated (and the matrix has not changed), then the `cacheSolve` should retrieve the inverse from the cache.

### `makeCacheMatrix`



## cacheSolve



The screenshot displays the RStudio environment with the following components:

- Console:** Contains the R code for defining the `makeCacheMatrix` and `cacheSolve` functions.
- Environment:** Shows the global environment with variables `TM`, `TV`, `VM`, and `x`.
- Functions:** Lists the functions `cacheSolve` and `makeCacheM...`.

```
> makeCacheMatrix <- function(x = matrix())
+ {
+   inv <- NULL
+   set <- function(y){
+     x <- y
+     inv <- NULL
+   }
+   get <- function() x
+   setInverse <- function(solveMatrix) inv <- solveMatrix
+   getInverse <- function() inv
+   list(set = set, get = get, setInverse = setInverse, getInverse = getInverse)
+ }
> cacheSolve <- function(x, ...)
+ {
+   inv <- x$getInverse()
+   if(!is.null(inv)){
+     message("getting cached data")
+     return(inv)
+   }
+   data <- x$get()
+   inv <- solve(data)
+   x$setInverse(inv)
+   inv
+ }
>
> |
```

**Environment:**

Variable	Value
TM	5
TV	5
VM	0.604069024110576
x	num [1:7] 2 3 4 5 6 7 8

**Functions:**

Function Name	Definition
cacheSolve	function (x, ...)
makeCacheM...	function (x = matrix())

The Windows taskbar at the bottom shows the time as 7:27 PM on 05-Aug-20.