Directions: MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

1. (30) The table below contains analog pairs between Python and R. Fill in the blanks.

<table>
<thead>
<tr>
<th>Python</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>list</td>
<td>vector</td>
</tr>
<tr>
<td>lambda function</td>
<td>blank (a)</td>
</tr>
<tr>
<td>dictionary</td>
<td>blank (b)</td>
</tr>
<tr>
<td>map</td>
<td>blank (c)</td>
</tr>
</tbody>
</table>

2. (15) Fill in the blank:

   ```r
   > f <- function(x) x^2
   > f
   function(x) x^2
   > print(f)
   function(x) x^2
   > p blank ( f )
   function(x) x^2
   ```

3. (55) A graph adjacency matrix consists of 0s and 1s, with a 1 at element (i,j) meaning there is a link from i to j. The function `haslinks(adj, target)` determines which vertices in the graph have links to a given set of target vertices. Here are examples:

   ```r
   > m
   [1,] 1 0 1 0 1
   [2,] 1 1 0 0 1
   [3,] 1 0 0 1 1
   [4,] 0 1 1 1 0
   [5,] 1 1 0 1 1
   > haslinks(m, c(1,4))
   [1] 3 5
   > haslinks(m,4)
   [1] 3 4 5
   > haslinks(m,1:2)
   [1] 2 5
   > haslinks(m, c(1,3,5))
   [1] 1
   ```

In the first call, for instance, we ask which vertices have links to both vertex 1 and vertex 4, and the function reports that vertices 3 and 5 (rows 3 and 5 in the matrix) have that property.

Fill in the blanks:

```r
haslinks <- function(adj, target) {
    canreach <- function(outlinks) {
        which1s <- which( blank (a) )
        tmp <- blank (b) (target, which1s)
        as.integer( blank (c) (tmp, target))
    }
    tmp1 <- apply( blank (d) )
    which(tmp1 == 1)
}
```
Solutions:
1.a anonymous function
1.b list
1.c apply
2. rint.function
3.
```r
haslinks <- function(adj, target) {
canreachtarget <- function(outlinks) {
    which1s <- which(outlinks == 1)
    tmp <- intersect(target, which1s)
    as.integer(setequal(tmp, target))
}
tmp1 <- apply(adj, 1, canreachtarget)
which(tmp1 == 1)
}
```