```
C
C
C---- Set up limits in grid units
C
      XTMP = (XYZLIM(2,1)-XYZLIM(1,1))* (XYZLIM(2,2)-XYZLIM(1,2))*
     +       (XYZLIM(2,3)-XYZLIM(1,3))
C
      IF ((XYZLIM(2,1)-XYZLIM(1,1)).LT.0.0000001) THEN
        XTMP = (XYZLIM(2,2)-XYZLIM(1,2))* (XYZLIM(2,3)-XYZLIM(1,3))
      ELSE IF ((XYZLIM(2,2)-XYZLIM(1,2)).LT.0.0000001) THEN
        XTMP = (XYZLIM(2,1)-XYZLIM(1,1))* (XYZLIM(2,3)-XYZLIM(1,3))
      ELSE IF ((XYZLIM(2,3)-XYZLIM(1,3)).LT.0.0000001) THEN
        XTMP = (XYZLIM(2,2)-XYZLIM(1,2))* (XYZLIM(2,1)-XYZLIM(1,1))
      END IF
C
      IF (XTMP.GT.5.0) THEN
C
C----  more than 5 "gridpoints" in map, limits were input in grid units
C
        XMIN = NINT(XYZLIM(1,1))
        XMAX = NINT(XYZLIM(2,1))
        YMIN = NINT(XYZLIM(1,2))
        YMAX = NINT(XYZLIM(2,2))
        ZMIN = NINT(XYZLIM(1,3))
        ZMAX = NINT(XYZLIM(2,3))
      ELSE
C
C----     limits were input as fractional limits
C
C   Need INT of Max values..
        XMIN = NINT(XYZLIM(1,1)*NX)
        XMAX =  INT(XYZLIM(2,1)*NX)
        YMIN = NINT(XYZLIM(1,2)*NY)
        YMAX =  INT(XYZLIM(2,2)*NY)
        ZMIN = NINT(XYZLIM(1,3)*NZ)
        ZMAX =  INT(XYZLIM(2,3)*NZ)
      END IF
C
```

```python
number = 5

# outer if statement
if (number >= 0):
    # inner if statement
    if number == 0:
      print('Number is 0')

    # inner else statement
    else:
        print('Number is positive')

# outer else statement
else:
    print('Number is negative')

# Output: Number is positive
```

```
#!/usr/local/bin/perl

$a = 100;
# check the boolean condition using if statement
if( $a  ==  20 ) {
   # if condition is true then print the following
   printf "a has a value which is 20\n";
} elsif( $a ==  30 ) {
   # if condition is true then print the following
   printf "a has a value which is 30\n";
} else {
   # if none of the above conditions is true
   printf "a has a value which is $a\n";
}
```

```
if (samplings[i] == 1) {
  max.of.mins.results[[1]] <- maxs.of.mins
} else {
  max.of.mins.results <- CreateSubmatrix(rmsd.matrix, samplings[i], F, T)
}

nofsamplings <- length(max.of.mins.results)
nofpoints    <- length(max.of.mins.results[[1]])

for (j in 1:nofsamplings) {
  max.of.mins.results[[j]] <- sort(max.of.mins.results[[j]])
}

max.rmsd     <- 0
for (j in 1:nofsamplings) {
  if (max.rmsd < max.of.mins.results[[j]][nofpoints]) {
```