" " test fails.

If at some stage the space bar is pressed, A$ will be assigned the character representing a space, and so the program will branch to line 80 and the loop will not be repeated.

But what will happen while the loop is repeating? Line 30 increments the value of R at each repetition of the loop. The first time through, R would be set to 1, the second time through it would be set to 1 + 1 and so on. When the loop has been broken out of by the test on A$, we could read R to see what we had counted up to.

Computers, however, operate very quickly and so R could be in the hundreds by the time we press the space bar. What would we do if we wanted values of R only between 1 and 10? Line 80 sets up another loop to enable us to test R and divide it by 10 if it is larger than 10. As long as R is larger than 10, the test in line 90 will fail, the value of Q will be reset to 0 and the loop will be repeated. Line 110 divides the value of R by 10 and a result is not printed out until the value of R has been reduced to a figure of less than 10. Line 30 ensures that the value of R can never be 0.

In theory, then, this program should produce a random number varying between 1 and 9 inclusive. But does it? The INT statement ensures that the decimal fractions have been removed, therefore the possible values of R would be 1,2,3,4,5,6,7,8,9. The average of these numbers is 5 (because their sum is 45, and 45 ÷ 9 = 5). Try it and see. You could do this by running the program a number of times, noting the value of R each time and then calculating the average. Alternatively, you could add some lines to the program to make it run, say, 100 times, adding the value of R to another variable S and then dividing S by 100.

When we tried this, we found that the average value of R was well below 5 and so the numbers could not have been random. It is instructive to consider why this could be.

The problem is that although BASIC is fast, it is not fast enough. The first loop lets the value of R increment until it reaches hundreds, or even thousands, before we press the space bar. Unless you make a deliberate effort to vary the amount of time elapsing between seeing the HIT THE SPACE-BAR prompt and actually pressing it, chances are you will press it after a fairly regular lapse of time. In this time, the value of R will probably have increased to several hundred.

The divisions that take place to reduce the value of R to a figure below 10 do not take place until after the space bar has been pressed. This means that R will almost always be in the low hundreds before the divisions take place and so the final value of R will tend to be low.

Is it possible to write a routine that overcomes this problem? The answer is yes, if we can make the counting process fast enough for our reaction time to the HIT THE SPACE-BAR prompt to be truly unpredictable. The solution is to make the test for the 'greater than upper limit' part of the first loop. Consider this program:

```
10 REM  GAME OF DICE -- MAIN PROGRAM
20 RANDOMIZE
30 REM  YOUR THROW
40 REM  GOSUB 'THROW' ROUTINE
50 GOSUB 300
60 LET M = D
70 PRINT "YOUR SCORE IS A"
80 REM  GOSUB 'SELECT' ROUTINE
90 GOSUB 390
100 PRINT
110 REM  COMPUTER'S THROW
120 REM  GOSUB 'THROW' ROUTINE
130 GOSUB 300
140 LET C = D
150 PRINT "THE COMPUTER'S SCORE IS A"
160 REM  GOSUB 'SELECT' ROUTINE
170 GOSUB 390
180 PRINT
190 REM  WHO WON?
200 IF M = C THEN LET S$ = "DRAW"
210 IF M > C THEN LET S$ = "YOU WON"
220 IF M < C THEN LET S$ = "COMPUTER WON"
230 REM  PRINT RESULT
240 PRINT S$
250 END
260 REM
270 REM
280 REM
290 REM
300 REM  RANDOM DICE THROW SUBROUTINE
310 REM
320 LET D = INT(10*RND)
330 IF D > 6 THEN GOTO 320
340 IF D < 1 THEN GOTO 320
350 RETURN
360 REM
370 REM
380 REM
390 REM  SELECT SUBROUTINE
400 REM
410 IF D = 1 THEN GOSUB 530
420 IF D = 2 THEN GOSUB 600
430 IF D = 3 THEN GOSUB 670
440 IF D = 4 THEN GOSUB 740
450 IF D = 5 THEN GOSUB 810
460 IF D = 6 THEN GOSUB 880
470 RETURN
480 REM
490 REM
500 REM
510 'GRAPHICS' SUBROUTINES
520 REM
530 PRINT "  _______  "
540 PRINT " |       | "
550 PRINT " |       | "
560 PRINT " |   *   | "
570 PRINT " |       | "
580 PRINT " |_______| "
590 RETURN
600 PRINT "  _______  "
610 PRINT " |       | "
620 PRINT " |      *| "
630 PRINT " |       | "
640 PRINT " |*      | "
650 PRINT " |_______| "
660 RETURN
670 PRINT "  _______  "
680 PRINT " |       | "
690 PRINT " |      *| "
700 PRINT " |   *   | "
710 PRINT " |*      | "
720 PRINT " |_______| "
730 RETURN
740 PRINT "  _______  "
750 PRINT " |       | "
760 PRINT " | *   * | "
770 PRINT " |       | "
780 PRINT " | *   * | "
790 PRINT " |_______| "
800 RETURN
810 PRINT "  _______  "
820 PRINT " |       | "
830 PRINT " | *   * | "
840 PRINT " |   *   | "
850 PRINT " | *   * | "
860 PRINT " |_______| "
870 RETURN
880 PRINT "  _______  "
890 PRINT " |       | "
900 PRINT " | *   * | "
910 PRINT " | *   * | "
920 PRINT " | *   * | "
930 PRINT " |_______| "
940 RETURN
```