I, and prints its value whenever you type P.

```
NEW
OK
10 IF GET$ = "P" THEN GOSUB 100
20 I = I + 1
30 GOTO 10
100 PRINT I;
110 RETURN
RUN
 57  113  181  193  201  305  369 ^C
BREAK AT LINE 20
OK
```

Since XYBASIC automatically removes the parity bit from any character it reads, you cannot GET$ any character with an ASCII value greater than 127 (7F hexadecimal). And of course you should not try to GET$ characters such as <control-C> which have special meanings to XYBASIC.

The function GET is similar to GET$, but returns the ASCII value of the typed character instead of a string consisting of the character. GET is described in Section 8 below.

STR$ and VAL

STR$ and VAL allow conversion between numeric values and strings, and are especially helpful when used with other string functions to reformat numerical output. STR$ (X) turns a numeric value into its string equivalent, returning the string of characters which Extended XYBASIC would PRINT as the value of X. For example, the following program converts an amount to dollars and cents, and then uses STR$ to convert the dollars and cents to strings which are PRINTed as part of the string S$.

```
NEW
OK
10 INPUT "Amount" X
20 D = INT (X)
30 C = 100 * (X - INT (X))
40 S$ = STR$ (D) + " dollars and" + STR$ (C) + " cents"
50 PRINT "$"; X; "is"; S$
60 GOTO 10
RUN
Amount? 2.25
$ 2.25 is 2 dollars and 25 cents
Amount? .37
$ .37 is 0 dollars and 37 cents
Amount? 11.95
$ 11.95 is 11 dollars and 95 cents
Amount? ^C
BREAK AT LINE 10
OK
```

Conversely, VAL converts a string representation of a number into its numeric value; that is, VAL (A$) gives the numeric value of the constant represented by the string A$. In the following program VAL finds the value of the amount typed after the dollar sign, ignoring the string preceding