

1. Intro. After a SAT solver has solved a problem set up with the SAT-LIFE programs, we want to see the answer in a convenient form. This program accepts the results (one line per solution) and converts the literals of the form *dad* into the rectangular “**dots**” format of periods and asterisks.

Input and output go from *stdin* to *stdout*.

```
#include <stdio.h>
#include <stdlib.h>
char pix[101][101];
⟨Subroutine 2⟩;
main()
{
    register int c, i, j, bit, maxi = 0, maxj = 0;
    while (1) {
        if (feof(stdin)) break;
        ⟨Process the next line of input 3⟩;
    }
}
```

2. ⟨Subroutine 2⟩ ≡

```
int nextchar(void)
{
    register int c = fgetc(stdin);
    if (c ≠ EOF) return c;
    exit(-1);
}
```

This code is used in section 1.

3. ⟨Process the next line of input 3⟩ ≡

```
for (c = nextchar(); c ≠ '␣'; ) {
    ⟨Process a literal 4⟩;
}
⟨Output the pixels found 6⟩;
```

This code is used in section 1.

4. $\langle \text{Process a literal } 4 \rangle \equiv$

```

c = nextchar();
if (c ≠ '~') bit = 1;
else {
    bit = 0;
    c = nextchar();
}
for (i = 0; c ≥ '0' ∧ c ≤ '9'; c = nextchar()) i = 10 * i + c - '0';
if (i ≥ 100) {
    fprintf(stderr, "Eh?_I_found_a_number_of_more_than_two_digits!\n");
    exit(-2);
}
if (c ≠ 'a') goto litdone;
c = nextchar();
for (j = 0; c ≥ '0' ∧ c ≤ '9'; c = nextchar()) j = 10 * j + c - '0';
if (j ≥ 100) {
    fprintf(stderr, "Eh?_I_found_a_number_of_more_than_two_digits!\n");
    exit(-2);
}
if (c ≠ '_' ∧ c ≠ '\n') goto litdone;
 $\langle \text{Record the pixel value } (i, j) \text{ } 5 \rangle$ ;
litdone: while (c ≠ '_' ∧ c ≠ '\n') c = nextchar();

```

This code is used in section 3.

5. $\langle \text{Record the pixel value } (i, j) \text{ } 5 \rangle \equiv$

```

if (i > maxi) maxi = i;
if (j > maxj) maxj = j;
pix[i][j] = bit;

```

This code is used in section 4.

6. $\langle \text{Output the pixels found } 6 \rangle \equiv$

```

for (i = 0; i ≤ maxi + 1; i++) {
    for (j = 0; j ≤ maxj + 1; j++) putchar(pix[i][j] ? '*' : '.');
    putchar('\n');
}
putchar('\n');

```

This code is used in section 3.

7. Index.

bit: [1](#), [4](#), [5](#).

c: [1](#), [2](#).

EOF: [2](#).

exit: [2](#), [4](#).

feof: [1](#).

fgetc: [2](#).

fprintf: [4](#).

i: [1](#).

j: [1](#).

litdone: [4](#).

main: [1](#).

maxi: [1](#), [5](#), [6](#).

maxj: [1](#), [5](#), [6](#).

nextchar: [2](#), [3](#), [4](#).

pix: [1](#), [5](#), [6](#).

putchar: [6](#).

stderr: [4](#).

stdin: [1](#), [2](#).

stdout: [1](#).

- ⟨Output the pixels found [6](#)⟩ Used in section [3](#).
- ⟨Process a literal [4](#)⟩ Used in section [3](#).
- ⟨Process the next line of input [3](#)⟩ Used in section [1](#).
- ⟨Record the pixel value (i, j) [5](#)⟩ Used in section [4](#).
- ⟨Subroutine [2](#)⟩ Used in section [1](#).

SAT-LIFE-FILTER

	Section	Page
Intro	1	1
Index	7	3