



```
*****  
* convolve.c  
***** /
```

```
/* Standard includes */  
#include <assert.h>  
#include <math.h>  
#include <stdlib.h> /* malloc(), realloc() */
```

```
/* Our includes */  
#include "base.h"  
#include "error.h"  
#include "convolve.h"  
#include "klt_util.h" /* printing */
```

```
#define MAX_KERNEL_WIDTH 71
```

```
typedef struct {  
    int width;  
    float data[MAX_KERNEL_WIDTH];  
} ConvolutionKernel;
```

```
/* Kernels */
```

Fundamentals of Programming

session 9

More on C

Example

Write a program which reads a number and prints its absolute value

Example

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    if (a >= 0) {
        printf("%d\n", a);
    }
    else {
        printf("%d\n", -a);
    }

    return 0;
}
```

Example

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    if (a >= 0) {
        printf("%d\n", a);
    }
    else {
        printf("%d\n", -a);
    }

    return 0;
}
```

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    if (a < 0) {
        a = -a;
    }

    printf("%d\n", a);

    return 0;
}
```

Example

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    if (a >= 0) {
        printf("%d\n", a);
    }
    else {
        printf("%d\n", -a);
    }

    return 0;
}
```

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    if (a < 0) {
        a = -a;
    }

    printf("%d\n", a);
    return 0;
}
```

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

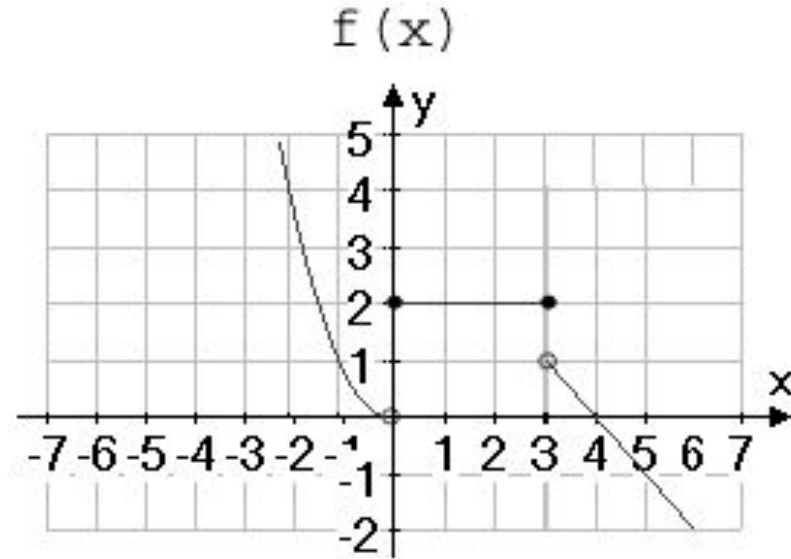
    if (a < 0)
        a = -a;

    printf("%d\n", a);

    return 0;
}
```

Piecewise functions

$$f(x) = \begin{cases} x^2, & x < 0 \\ 2, & 0 \leq x \leq 3 \\ 4 - x, & x > 3 \end{cases}$$



http://math2.uncc.edu/~bjwichno/spring2010/math1121/Lecture_Notes/unit_1/Lectures/lec_piecewise_rule.htm

Piecewise functions

$$f(x) = \begin{cases} x^2, & x < 0 \\ 2, & 0 \leq x \leq 3 \\ 4 - x, & x > 3 \end{cases}$$

```
float x,y;
```

```
scanf("%f", &x);
```

```
if (x < 0)
```

```
    y = x*x;
```

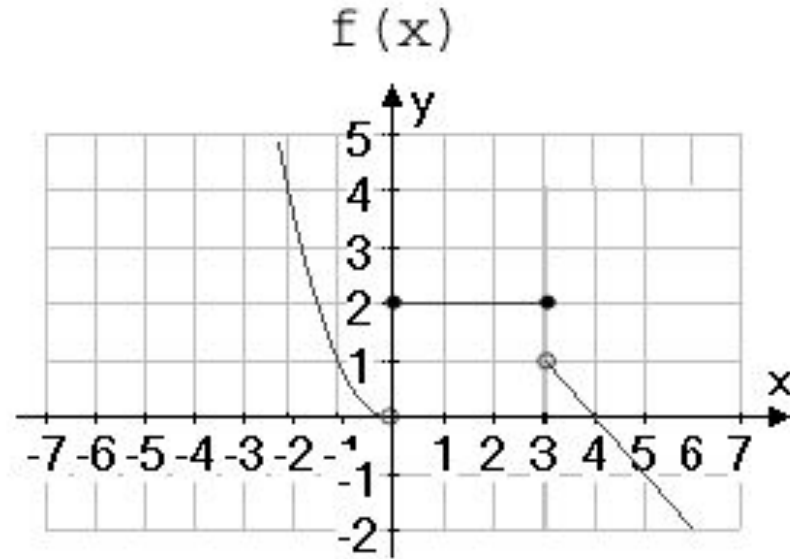
```
else if (x <= 3)
```

```
    y = 2;
```

```
else
```

```
    y = 4-x;
```

```
printf("%f\n", y);
```



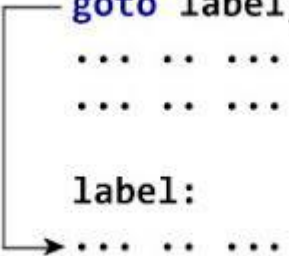
http://math2.uncc.edu/~bjwichno/spring2010/math1121/Lecture_Notes/unit_1/Lectures/lec_piecewise_rule.htm

Loops



goto

```
goto label;  
... ..  
... ..  
  
label:  
... ..  
... ..
```

A diagram illustrating the goto statement. A blue 'goto label;' statement is shown at the top. Below it are two lines of three dots each. Further down is a label 'label:' followed by two lines of three dots each. A black line starts from the end of the 'goto' statement, goes down, then right, then down again, ending in an arrowhead pointing to the first line of code under the 'label:'.

```
#include <stdio.h>

int main() {
    int N;
    int i;

    scanf("%d",&N);

    i = 1;

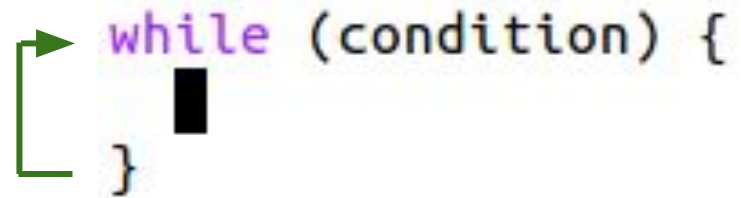
l1:
    if (i > N)
        goto l2;
    printf("%d\n", i);
    i = i + 1;

    goto l1;
l2:

    return 0;
}
```

while loop

```
while (condition) {  
    █  
}
```

A diagram illustrating the structure of a while loop. The code is shown as `while (condition) {` on the first line, followed by a black square representing a code block on the second line, and `}` on the third line. A green arrow starts from the left side of the opening curly brace, goes up, then right, then down, and finally right again, pointing to the opening curly brace of the loop body, indicating the flow of execution.

```
#include <stdio.h>

int main() {
    int N;
    int i;

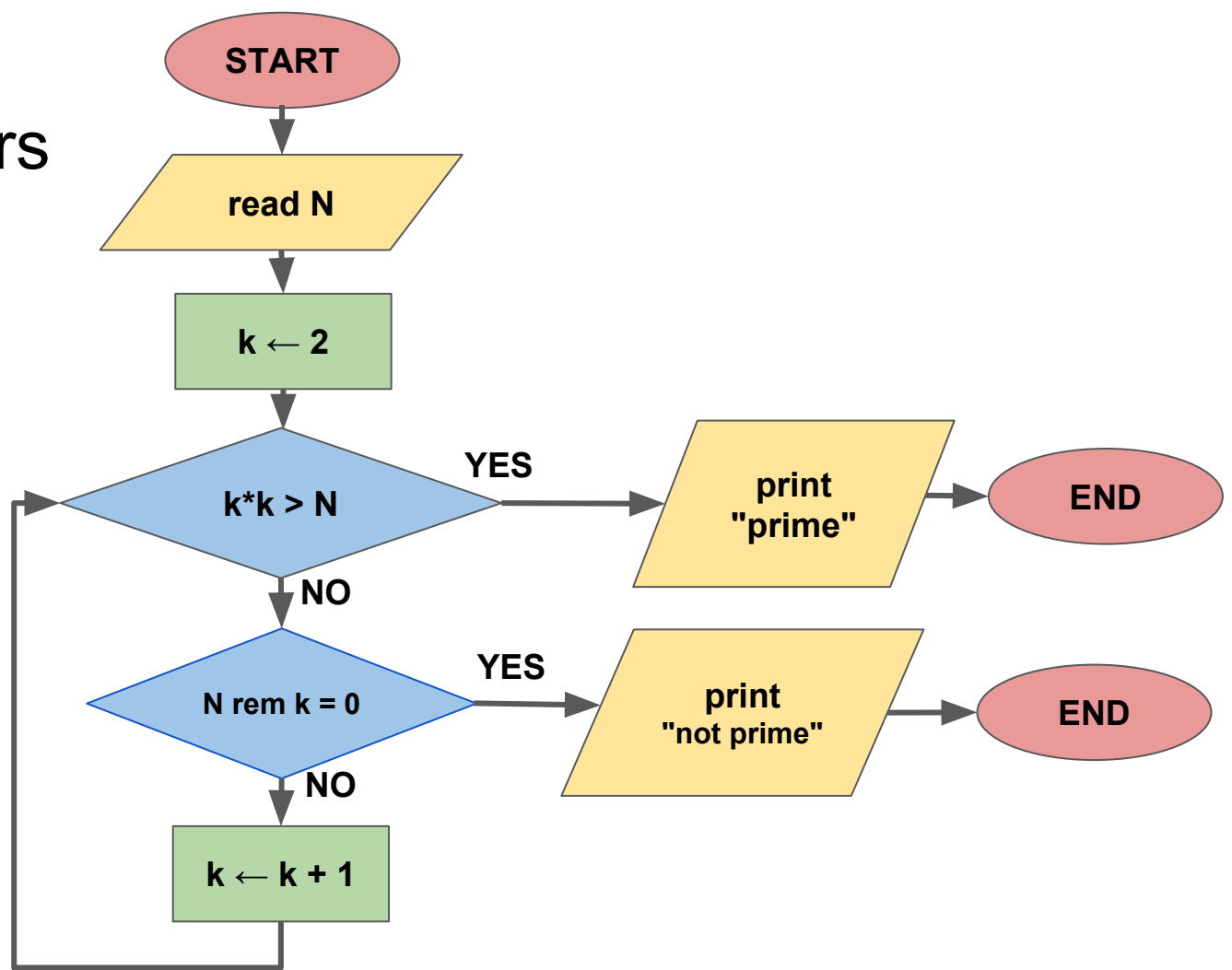
    scanf("%d",&N);

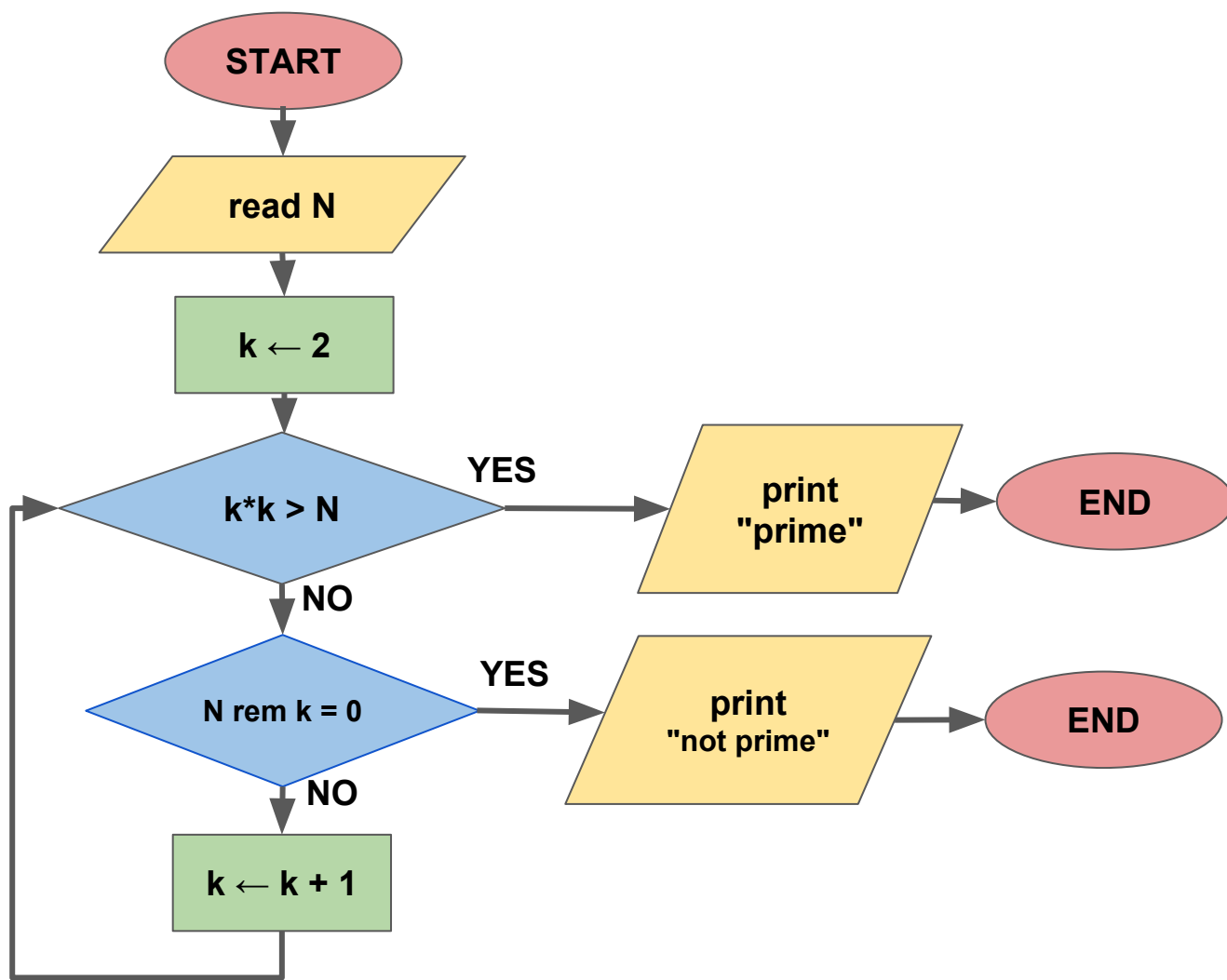
    i = 1;

    while (i <= N) {
        printf("%d\n", i);
        i = i + 1;
    }

    return 0;
}
```

Prime numbers





```

unsigned int N,k;

scanf("%d", &N);

if (N == 1) {
    printf("Not Prime\n");
    goto end;
}

k = 2;

start:
if (k*k > N) {
    printf("Prime\n");
    goto end;
}

if (N % k == 0) {
    printf("Not Prime\n");
    goto end;
}

k = k + 1;

goto start;

end:

return 0;
  
```

Avoid using the goto command!

```
unsigned int N,k;

scanf("%d", &N);

if (N == 1) {
    printf("Not Prime\n");
    goto end;
}

k = 2;

start:
if (k*k > N) {
    printf("Prime\n");
    goto end;
}

if (N % k == 0) {
    printf("Not Prime\n");
    goto end;
}

k = k + 1;

goto start;

end:

return 0;
```

```
#include <stdio.h>

int main() {
    unsigned int N,k;

    scanf("%d", &N);

    k = 2;

    while (k*k <= N && N % k != 0) {
        k = k + 1;
    }

    if (k*k > N)
        puts("Prime");
    else
        puts("Not Prime");
}
```



```
#include <stdio.h>

int main() {
    unsigned int N,k;

    scanf("%d", &N);

    k = 2;

    while (k*k <= N && N % k != 0) {
        k = k + 1;
    }

    if (k*k > N)
        puts("Prime");
    else
        puts("Not Prime");
}
```

```
#include <stdio.h>

int main() {
    unsigned int N,k, prime;
    scanf("%d", &N);

    prime = 1;
    k = 2;
    while (k*k <= N) {

        if (N % k == 0)
            prime = 0;

        k = k + 1;
    }

    if (prime == 1)
        puts("Prime");
    else
        puts("Not Prime");
}
```

```
#include <stdio.h>

int main() {
    unsigned int N,k, prime;
    scanf("%d", &N);

    prime = 1;
    k = 2;
    while (k*k <= N) {

        if (N % k == 0)
            prime = 0;

        k = k + 1;
    }

    if (prime == 1)
        puts("Prime");
    else
        puts("Not Prime");
}
```

```
#include <stdio.h>

int main() {
    unsigned int N,k, prime;
    scanf("%d", &N);

    prime = 1;
    k = 2;
    while (k*k <= N) {

        if (N % k == 0) {
            prime = 0;
            break;
        }

        k = k + 1;
    }

    if (prime == 1)
        puts("Prime");
    else
        puts("Not Prime");
}
```