CS Summer Challenge

Day 0x4

How does code run?

In These Slides

- Learn about *‘the for loop’* programming structure.
  - Used to repeat instructions for a known number of repetitions.
- For loops make source code shorter, removing a lot of redundancy that results from repeating the same instructions multiple times.
  - The compiler or interpreter (a.k.a. translator software) translates a for loop back into a long sequence of repeated binary instructions, run one after the other by the CPU.
- Learn about *‘the stack’*: function-specific memory.
  - Used to store variables local to a function.
  - Those are variables created by the function and used by it and by no other function.
- When a function begins running, a region of its binary code memory is reserved for its local variables.
  - When the function returns (a.k.a. ends), the stack memory is deleted, therefore preventing any other function from using it.

For Loops: Motivation

- How can we write the ***factorial*** math function in code?
  - factorial(5) = 5 x 4 x 3 x 2 x 1
  - factorial(11) = 11 x 10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
  - factorial(N) = N x (N - 1) x (N - 2) x … x 1
  - Without a for loop:

```
int factorial = 1;
int i = N;
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
          ...
```

For Loops: Motivation

- How can we write the ***factorial*** math function in code?
  - factorial(5) = 5 x 4 x 3 x 2 x 1
  - factorial(11) = 11 x 10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
  - factorial(N) = N x (N - 1) x (N - 2) x … x 1
  - Without a for loop:

i = 5
*factorial = 1 * 5 = 5*

i = 5 - 1 = 4
*factorial = 5 * 4 = 20*

i = 4 - 1 = 3
*factorial = 20 * 3 = 60*

i = 3 - 1 = 2
...

```
int factorial = 1;
int i = N;
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
factorial = factorial * i;
if ( i < 1 ) return; // stop here
else { i = i - 1; }
```

...

For Loops: Motivation

- How can we write the ***factorial*** math function in code?
  - factorial(5) = 5 x 4 x 3 x 2 x 1
  - factorial(11) = 11 x 10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
  - factorial(N) = N x (N - 1) x (N - 2) x … x 1
  - With a for loop:

C For Loop

```
int factorial = 1;
int i;
for ( i = N; i > 0; i = i - 1) {
     factorial = factorial * i;
}
```

For Loops: Motivation

- How can we write the ***factorial*** math function in code?
  - factorial(5) = 5 x 4 x 3 x 2 x 1
  - factorial(11) = 11 x 10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
  - factorial(N) = N x (N - 1) x (N - 2) x … x 1
  - With a for loop:

Python For Loop

```
factorial = 1
for i in range(N, 0, -1):
    factorial = factorial * i
```

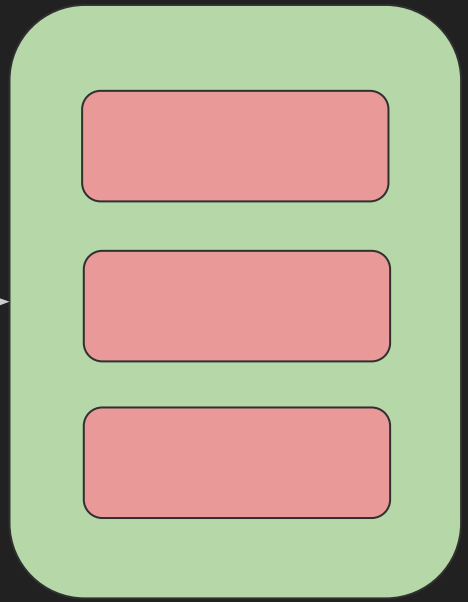
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000001010100010101000
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010011010
01000111000100101010000000
00001010001010100100000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor



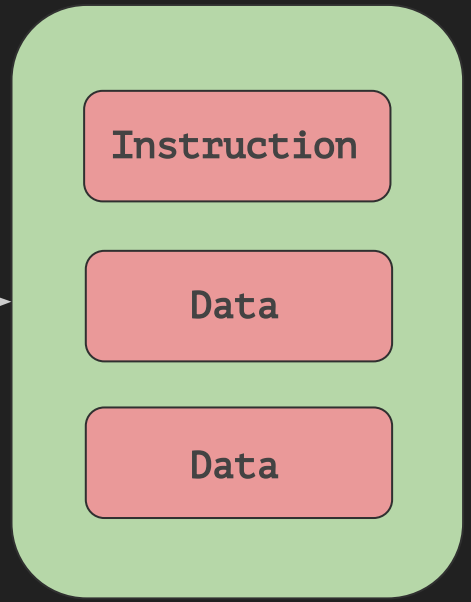
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000001010100010101000
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010011010
01000111000100101010000000
00001010001010100100000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor



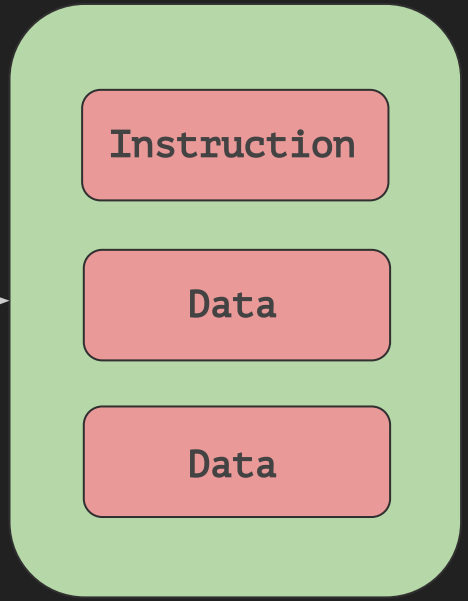
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

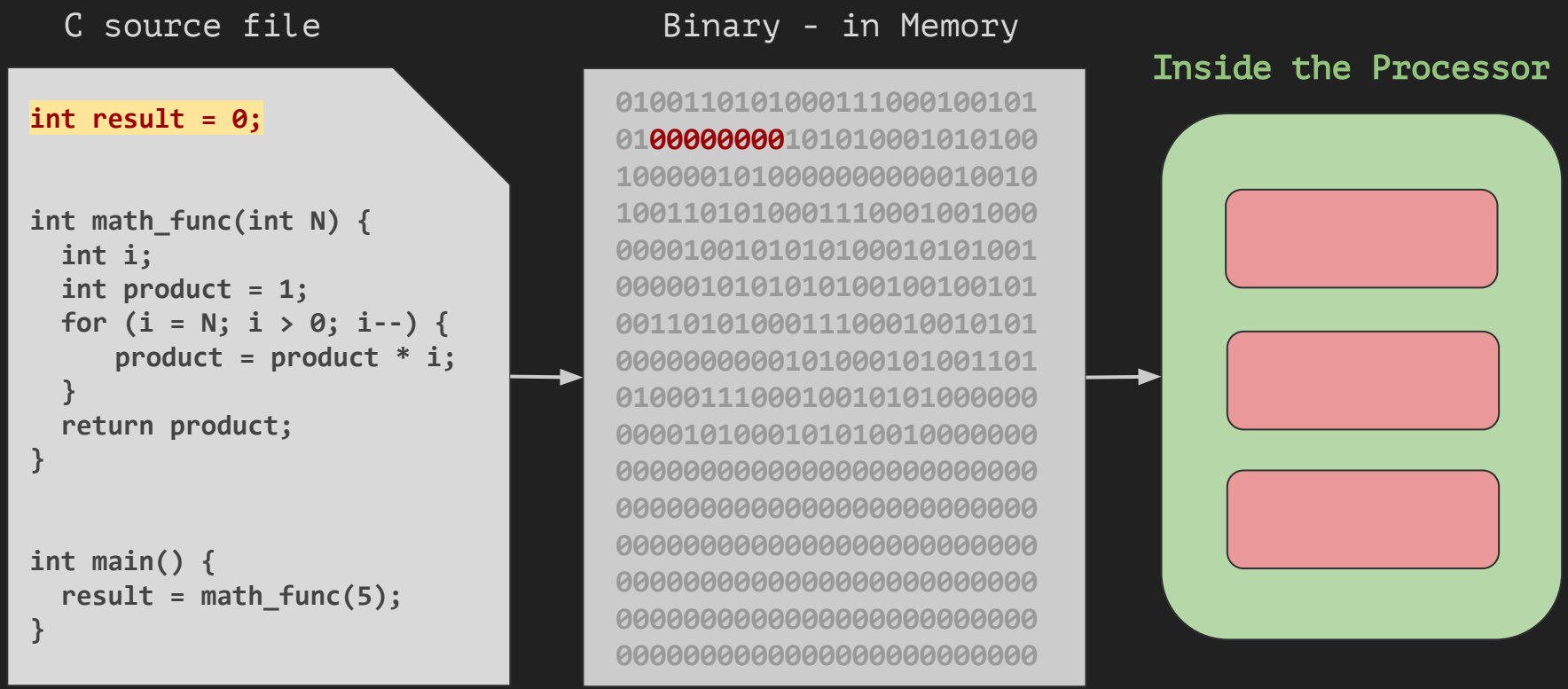
Binary - in Memory

```
01001101010001110001001010
01000000001010100010101000
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010011010
01000111000100101010000000
00001010001010100100000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor



Watch what happens as the code executes...


C source file
int result = 0;
int math_func(int N) {
int i;
int product = 1;
for (i = N; i > 0; i--) {
product = product * i;
}
return product;
}
int main() {
result = math_func(5);
}
Binary - in Memory
01001101010001110001001010
01000000000101010001010100
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010011010
01000111000100101010000000
00001010001010100100000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
Inside the Processor

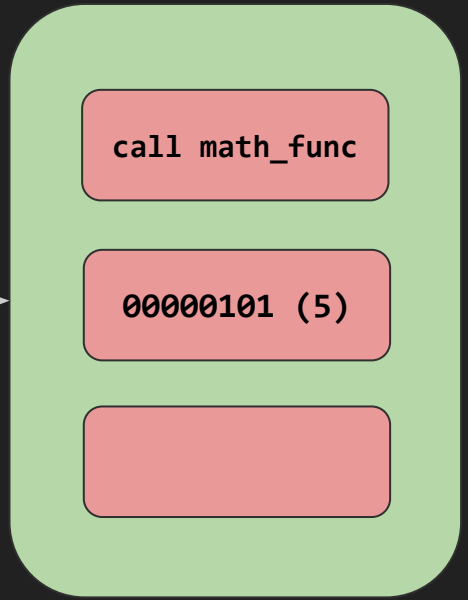
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor



Execution starts in main...

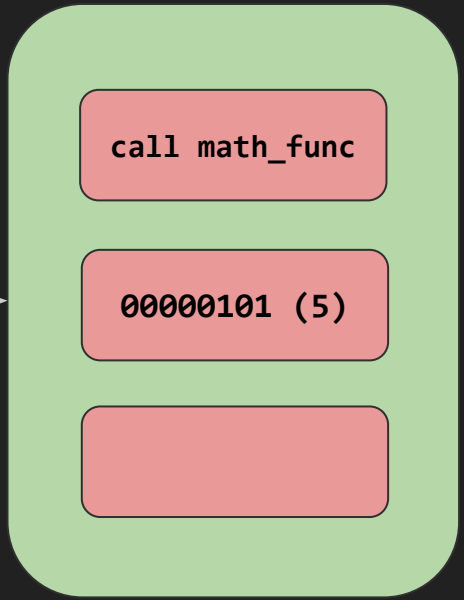
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor



math_func
stack memory

Calling math_func...

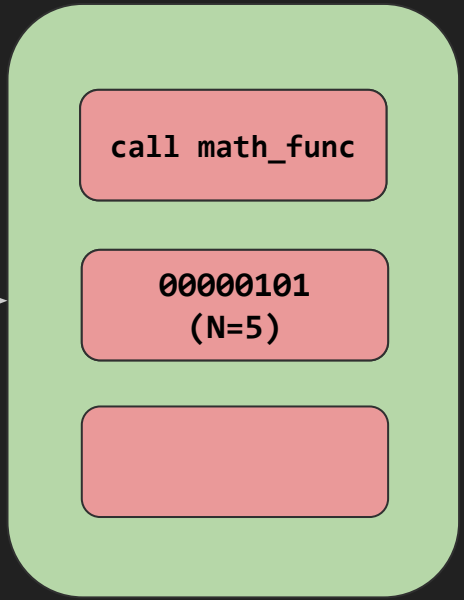
C source file

```
int result = 0;


int math_func(int N) {
  int i;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor



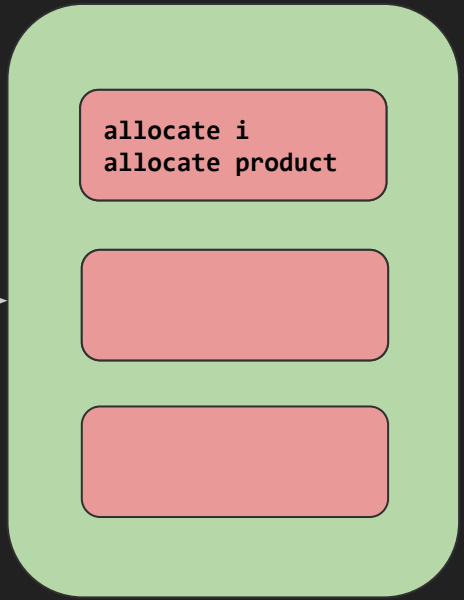
C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000000101010001010100
10000010100000000010010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000000101000000000000000
00000000000000000000010000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor



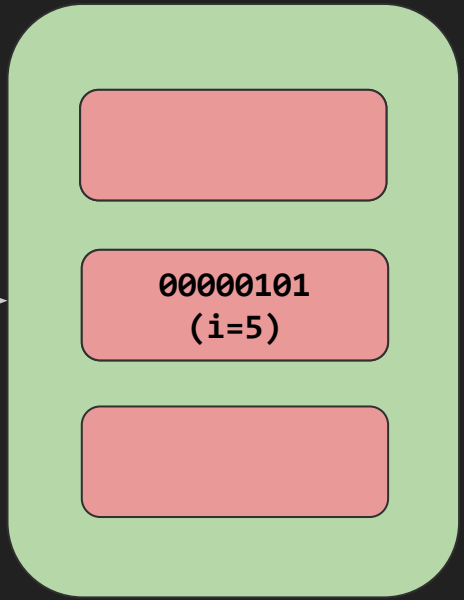
C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

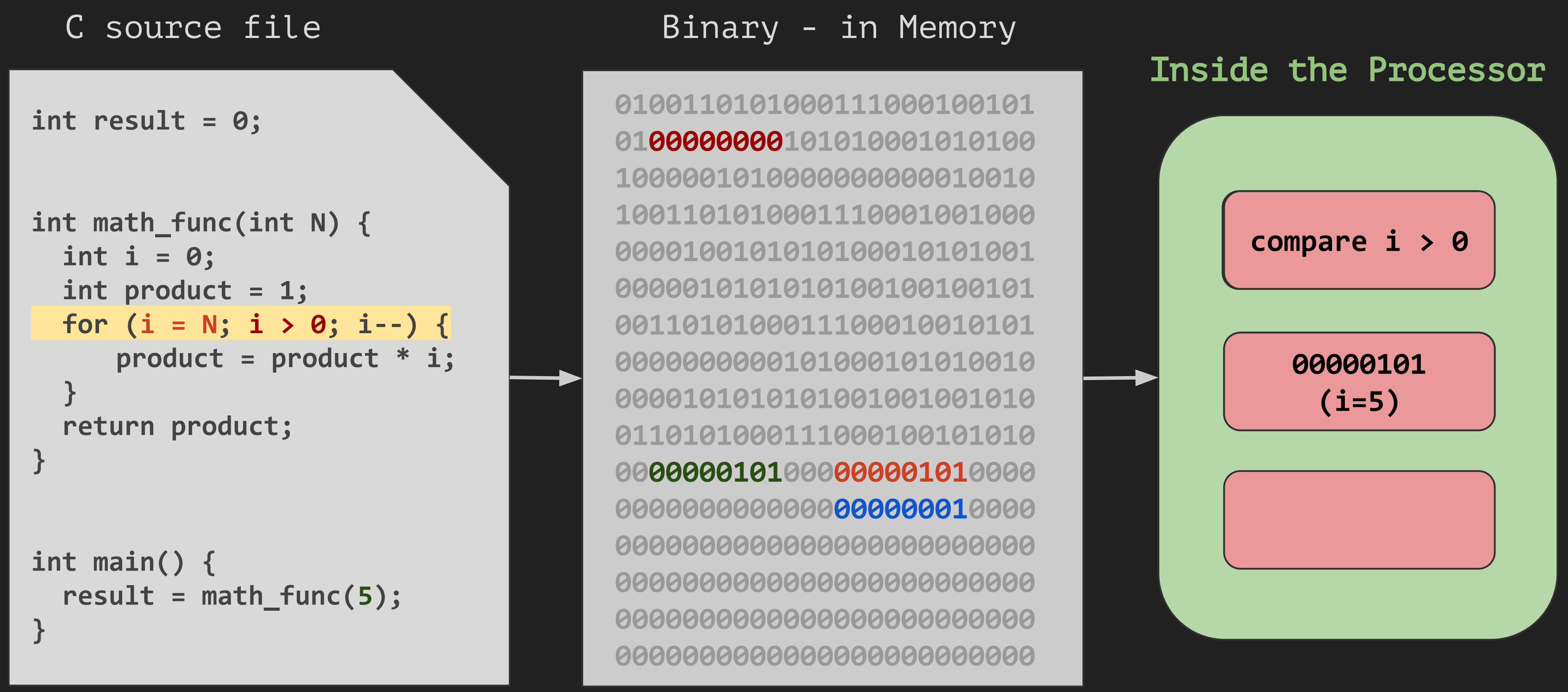
Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000010100000
0000000000000000001000010
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor



for loop begins...



C source file
int result = 0;
int math_func(int N) {
int i = 0;
int product = 1;
for (i = N; i > 0; i--) {
product = product * i;
}
return product;
}
int main() {
result = math_func(5);
}
Binary - in Memory
0100110101000111000100101
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000001010000
0000000000000000000010000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
Inside the Processor
compare i > 0
00000101
(i=5)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000001010100010101000
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000010100000000010100000
00000000000000000010100000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor

multiply

00000101
(i=5)

00000101
(product=5)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000100000000
0000000000000000101000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

decrement i

00000100
(i=4)

00000101
(product=5)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000000101010001010100
10000010100000000010010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000000101000000010000000
00000000000000000010100000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor

compare i > 0

00000100
(i=4)

00000101
(product=5)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(57);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000100000000
0000000000000010100000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

multiply

00000100
(i=4)

00010100
(product=20)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000001100000
0000000000000000010100000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

decrement i

00000011
(i=3)

00010100
(product=20)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

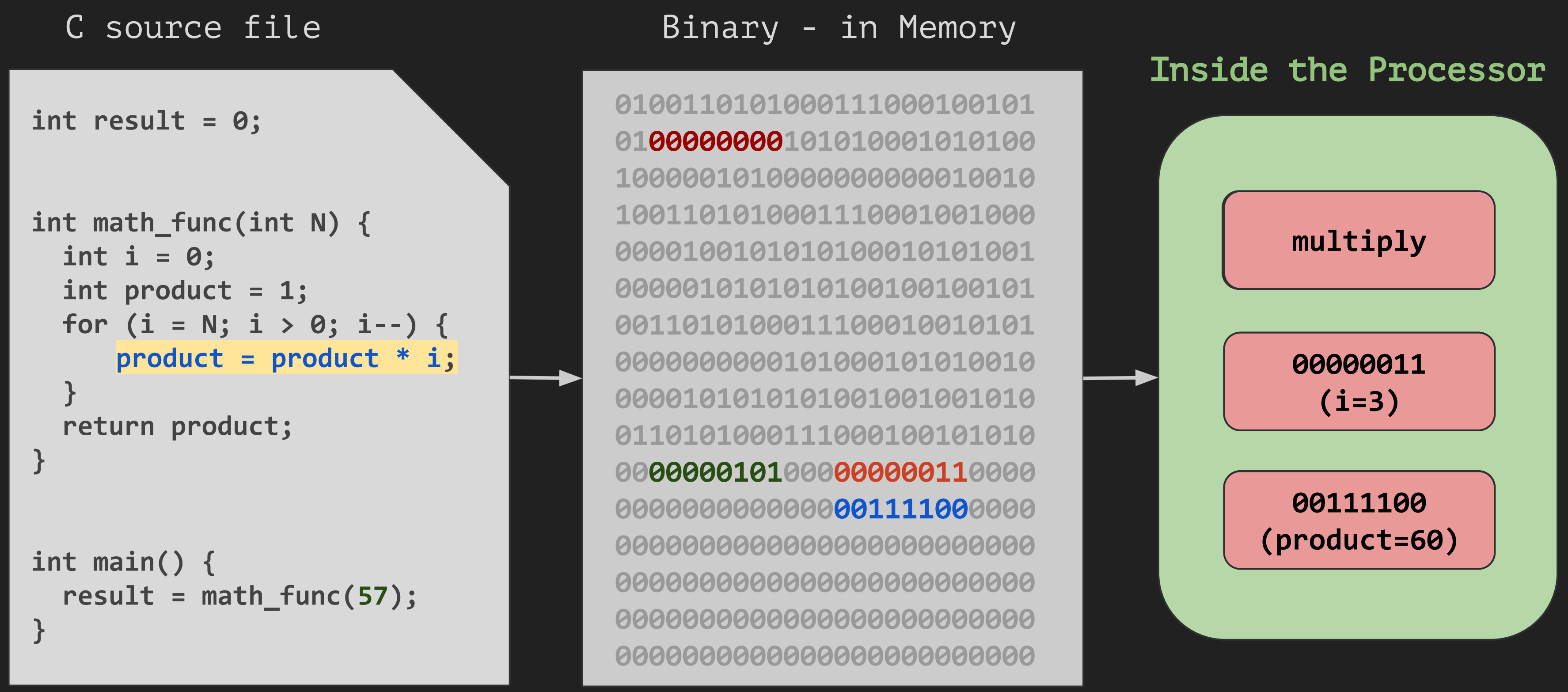
```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000001100000
0000000000000010100000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

compare i > 0

00000011
(i=3)

00010100
(product=20)


C source file
int result = 0;
int math_func(int N) {
int i = 0;
int product = 1;
for (i = N; i > 0; i--) {
product = product * i;
}
return product;
}
int main() {
result = math_func(57);
}
Binary - in Memory
0100110101000111000100101
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000110000
0000000000000001111000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
Inside the Processor
multiply
00000011
(i=3)
00111100
(product=60)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000001000000
0000000000000001111000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

decrement i

00000010
(i=2)

00111100
(product=60)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000000101010001010100
10000010100000000010010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000001010000000010000000
00000000000000111100000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor

compare i > 0

00000010
(i=2)

00111100
(product=60)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(57);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000001010100010101000
10000010100000000000010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000010100000000001000000
00000000000001111000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor

multiply

00000010
(i=2)

01111000
(product=120)

C source file

```c
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

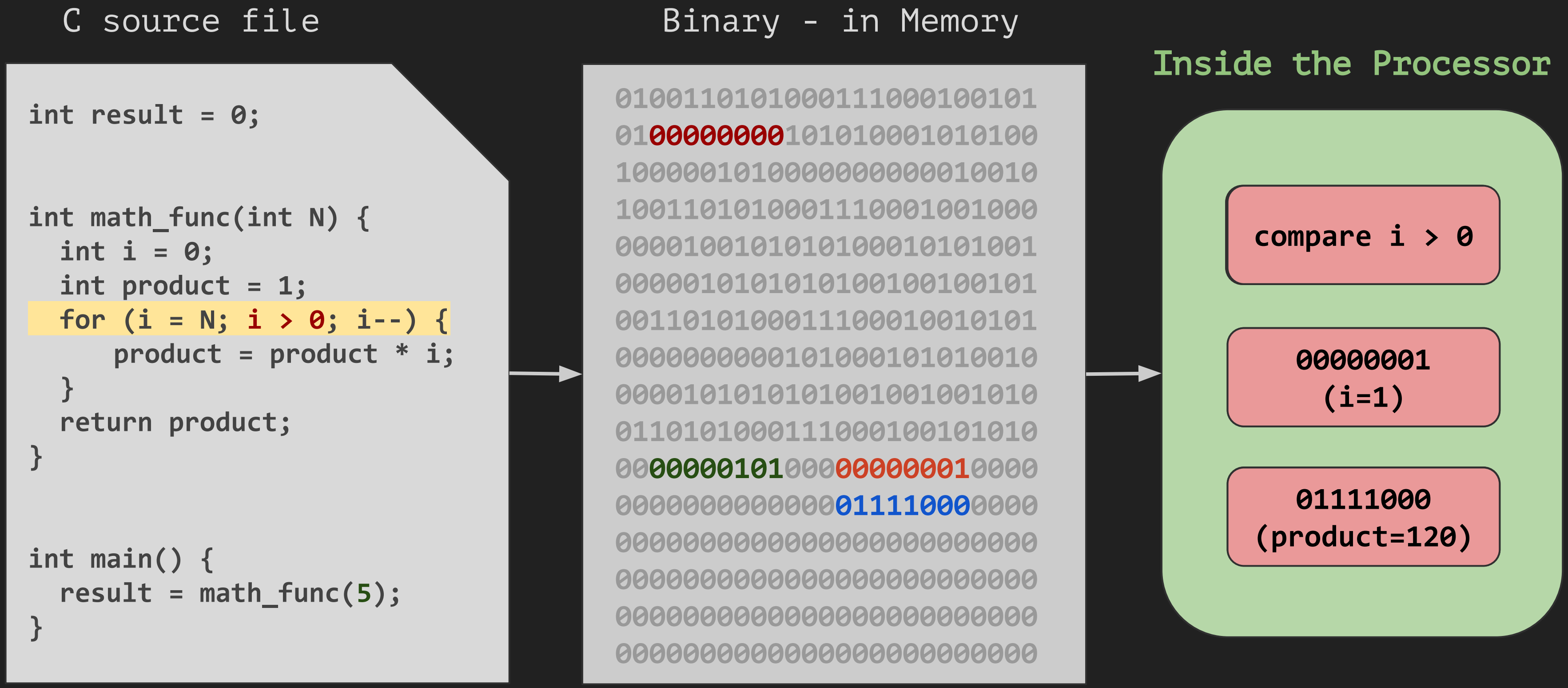
```
0100110101000111000100101
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000010000
0000000000000011110000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

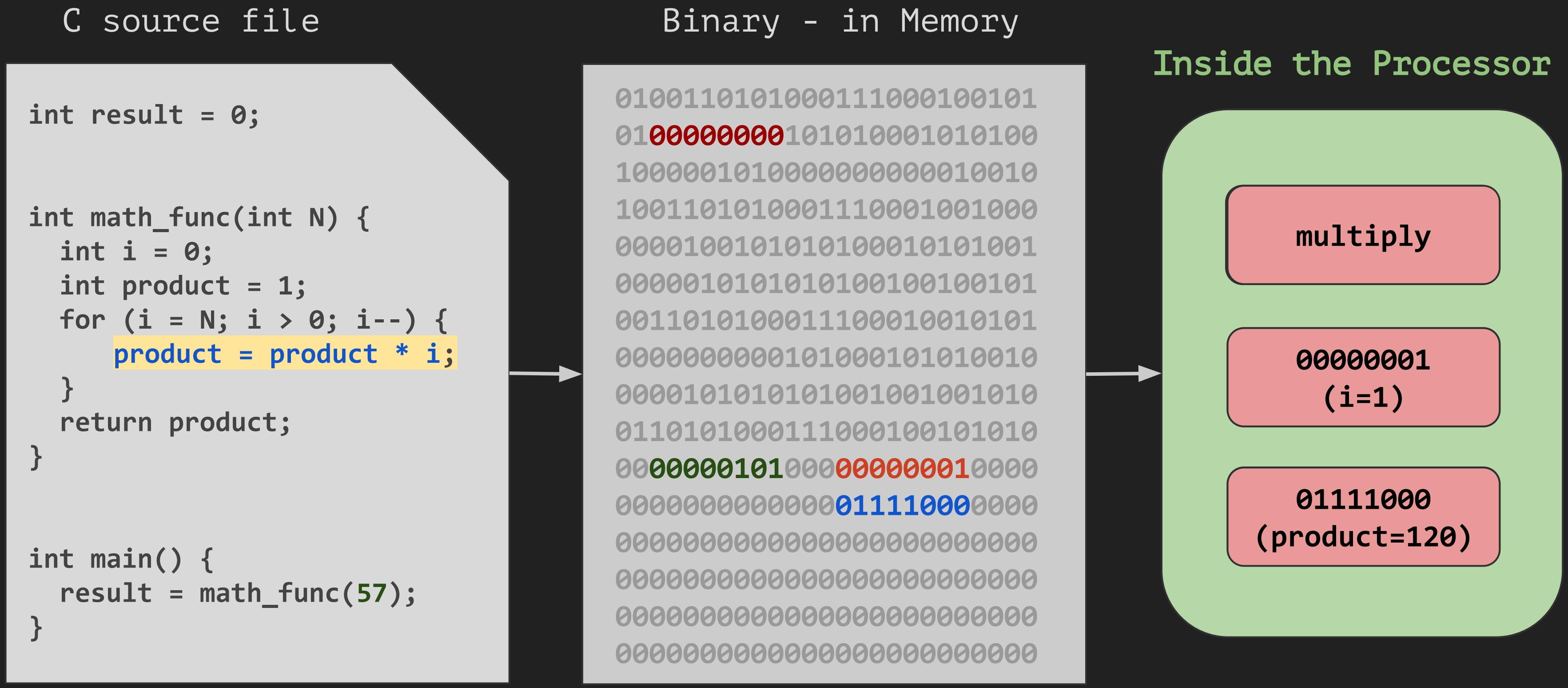
Inside the Processor

decrement i

00000001
(i=1)

01111000
(product=120)


C source file
int result = 0;
int math_func(int N) {
int i = 0;
int product = 1;
for (i = N; i > 0; i--) {
product = product * i;
}
return product;
}
int main() {
result = math_func(5);
}
Binary - in Memory
0100110101000111000100101
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000010000
0000000000000011110000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
Inside the Processor
compare i > 0
00000001
(i=1)
01111000
(product=120)


C source file
int result = 0;
int math_func(int N) {
int i = 0;
int product = 1;
for (i = N; i > 0; i--) {
product = product * i;
}
return product;
}
int main() {
result = math_func(57);
}
Binary - in Memory
0100110101000111000100101
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000010000
0000000000000011110000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
Inside the Processor
multiply
00000001
(i=1)
01111000
(product=120)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000000000
0000000000000001111000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

decrement i

00000000
(i=0)

01111000
(product=120)

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0100000000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000101000000000000000
0000000000000001111000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor

compare i > 0

00000000
(i=0)

01111000
(product=120)

for loop ends...

C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
01000000000101010001010100
10000010100000000010010010
10011010100011100010010000
00001001010101000101010010
00000101010101001001001010
00110101000111000100101010
00000000001010001010100100
00001010101010010010010100
01101010001110001001010100
00000000101000000000000000
00000000000000011110000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
```

Inside the Processor

return from math_func back to main

00000000 (i=0)

01111000 (product=120)

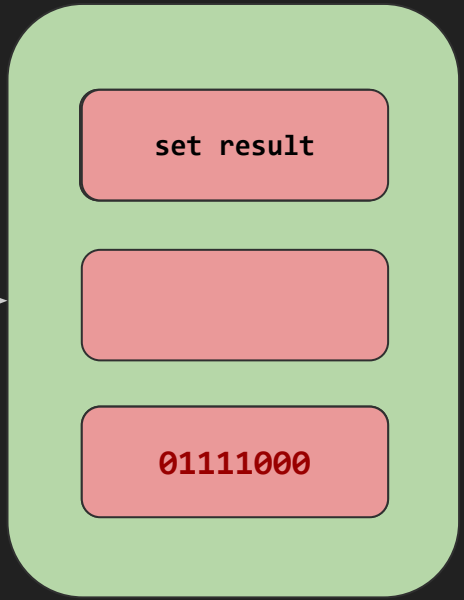
C source file

```
int result = 0;


int math_func(int N) {
  int i = 0;
  int product = 1;
  for (i = N; i > 0; i--) {
      product = product * i;
  }
  return product;
}


int main() {
  result = math_func(5);
}
```

Binary - in Memory

```
01001101010001110001001010
0101111000101010001010100
1000001010000000000010010
1001101010001110001001000
0000100101010100010101001
0000010101010100100100101
0011010100011100010010101
0000000000101000101010010
0000101010101001001001010
0110101000111000100101010
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
0000000000000000000000000
```

Inside the Processor



math_func returns... math_func memory is cleared...