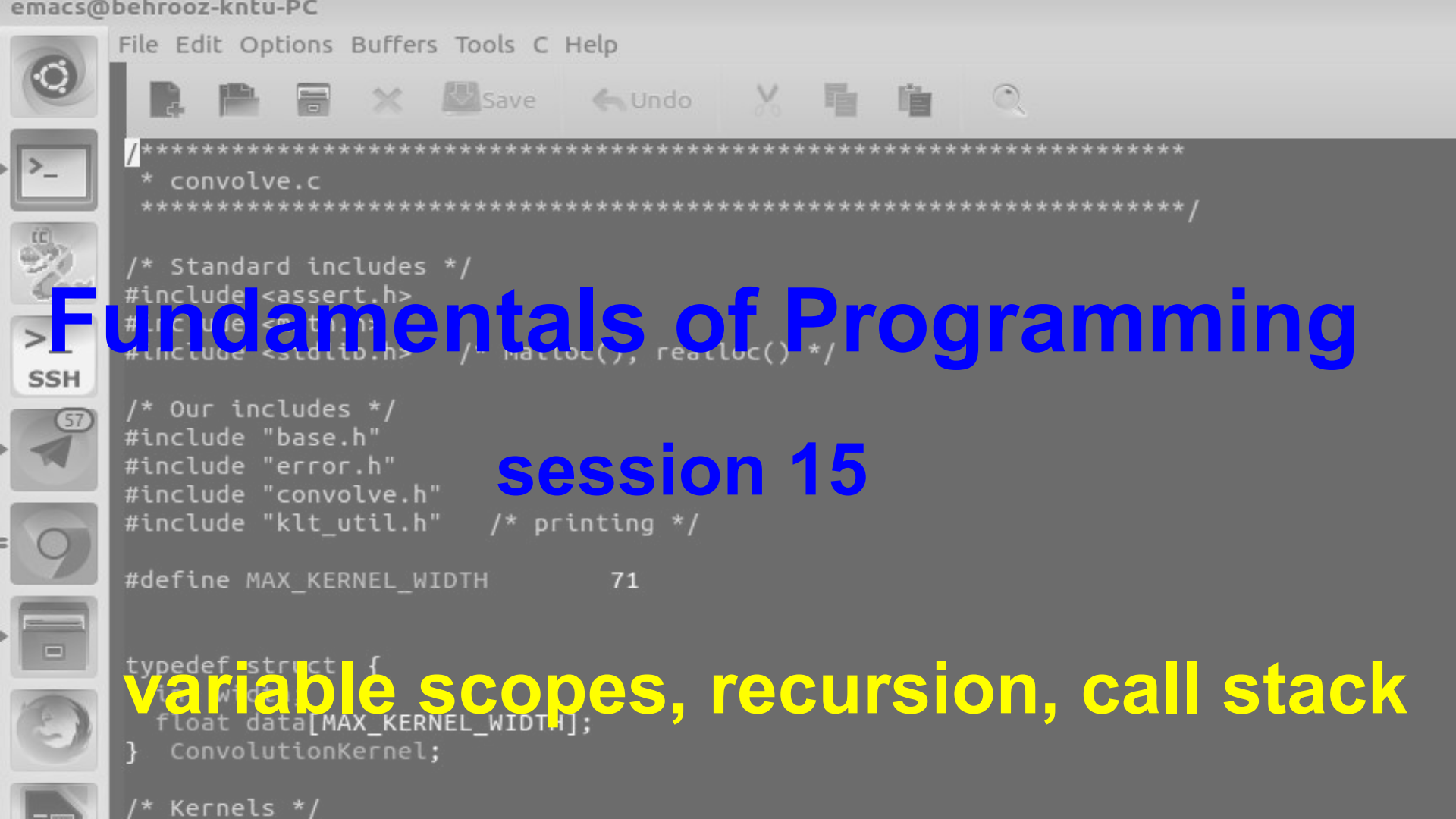
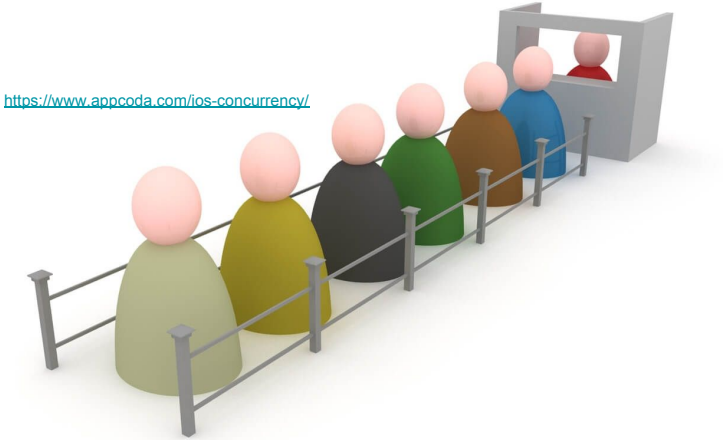

emacs@behrooz-kntu-PC
File Edit Options Buffers Tools C Help
Save
Undo
/* convolve.c */
/* Standard includes */
#include <assert.h>
/* Our includes */
#include "base.h"
#include "error.h"
#include "convolve.h"
#include "klt_util.h" /* printing */
#define MAX_KERNEL_WIDTH 71
typedef struct {
float data[MAX_KERNEL_WIDTH];
} ConvolutionKernel;
/* Kernels */
SSH
Fundamentals of Programming
session 15
variable scopes, recursion, call stack

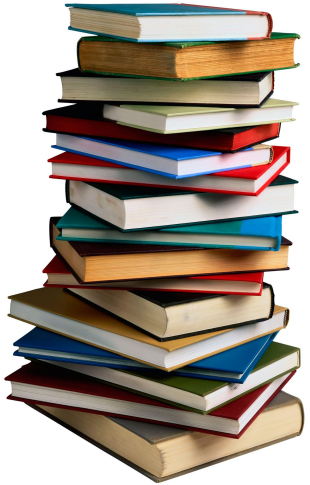
Queue vs Stack

https://www.appcoda.com/ios-concurrency/



queue

http://wikiclipart.com/stack-of-books-clipart_22617/



stack

Push and Pop

Variable scope

```c
#include <stdio.h>

int main() {
  int x = 2;

  if (1) {

    x = 3;

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

  }

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

}
```

Variable scope

```c
#include <stdio.h>

int main() {
  int x = 2;

  if (1) {

    x = 3;

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

  }

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

}
```

```c
#include <stdio.h>

int main() {
  int x = 2;

  if (1) {
    int x;
    x = 3;

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

  }

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

}
```

Variable scope

```c
#include <stdio.h>

int main() {
  int x = 2;

  if (1) {

    x = 3;

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

  }

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

}
```

```c
#include <stdio.h>

int main() {
  int x = 2;

  if (1) {
    int x;
    x = 3;

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

  }

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

}
```

```c
#include <stdio.h>

int main() {
  int x = 2;

  {
    int x;
    x = 3;

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

  }

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

}
```

Variable scope

```c
#include <stdio.h>

int main() {
  int a, x;

  {
    double b, x;

    {
      int x;
      int y;

    }


  }


}
```

Variable scope

```c
#include <stdio.h>

int main() {
  int a, x;

  {
    double b, x;

    {
      int x;
      int y;

    }


  }


}
```

```c
#include <stdio.h>

int main() {
  int a, x;

  a = 1;
  while (a < 10) {
    int x = 2;

    if (a > 5) {
      double a,x,d;

      printf("%f",x);
    }

    a++;
  }


}
```

Variable scope

```c
#include <stdio.h>

int main() {
  int a, x;

  {
    double b, x;

    {
      int x;
      int y;

    }


  }


}
```

```c
#include <stdio.h>

int main() {
  int a, x;

  a = 1;
  while (a < 10) {
    int x = 2;

    if (a > 5) {
      double a,x,d;

      printf("%f",x);
    }

    a++;
  }


}
```

```c
#include <stdio.h>

void func1(int a, double x, double d);

int main() {
  int a, b, x, y;

  a = 1;
  while (a < 10) {
    int x = 2;

    if (a > 5) {
      double x=2.0,d;
      func1(a,x,d);
    }

    a++;
  }
}

void func1(int a, double x, double d) {
  double y;
  double b;

  if (x > 0) {
    int b;
    y = x * d;
    b = (int) (a * x);

    printf("%d", b);
  }
}
```

Global variables

```c
#include <stdio.h>

void sqr(void);

int x;


int main() {
  x = 4;

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

  sqr();

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

}


void sqr() {
  x = x*x;

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

}
```

Global variables

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  x = x*x;

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

}
```

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  int x;

  x = x * x;

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

}
```

Global variables

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  x = x*x;

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

}
```

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  int x;

  x = x * x;

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

}
```

```c
#include <stdio.h>

void sqr(int);

int x;

int main() {
  x = 4;

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

  sqr(x);

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

}

void sqr(int x) {
  x = x*x;

  printf("%d\n",x);
}
```

Global variables

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  x = x*x;

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

}
```

```c
#include <stdio.h>

void sqr(void);

int x;

int main() {
  x = 4;

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

  sqr();

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

}

void sqr() {
  int x;

  x = x * x;

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

}
```

```c
#include <stdio.h>

void sqr(int);

int x;

int main() {
  x = 4;

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

  sqr(x);

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

}

void sqr(int x) {
  x = x*x;

  printf("%d\n",x);
}
```

Recursion

factorial function:

- `n! = 1 * 2 * … * n`

Recursion

factorial function:

- `n! = 1 * 2 * … * n`

```c
#include <stdio.h>

int fact(int);


int main() {
  int n;

  scanf("%d", &n);

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

}


int fact(int n) {
  int p = 1;

  for (int i = 2; i <= n; i++)
    p *= i;

  return p;

}
```

Recursion

recursive definition:

- `0! = 1`
- `n! = n * (n-1)!`

Recursion

recursive definition:

- `0! = 1`
- `n! = n * (n-1)!`

```c
#include <stdio.h>

int fact(int);


int main() {
  int n;

  scanf("%d", &n);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  // ???
```

Recursion

recursive definition:

- `0! = 1`
- `n! = n * (n-1)!`

```c
#include <stdio.h>

int fact(int);


int main() {
  int n;

  scanf("%d", &n);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

Recursion

recursive definition:

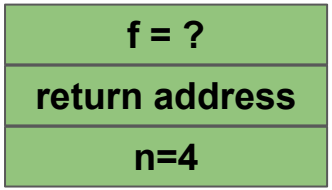
- `0! = 1`
- `n! = n * (n-1)!`

```c
#include <stdio.h>

int fact(int);

int main() {
  int n;

  scanf("%d", &n);

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

}


int fact(int n) {
  if (n == 0)
    return 1;

  return fact(n-1) * n;

}
```

Call stack

```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;

  printf("%d\n", fact(m));

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

stack frame:

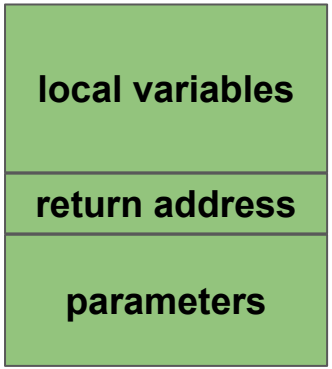


Call stack

```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;

  printf("%d\n", fact(m));

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

stack frame:

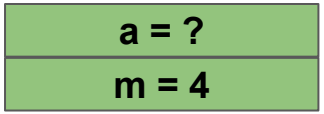
- **parameters (arguments)**
- **return address**
- **local variables**



Call stack

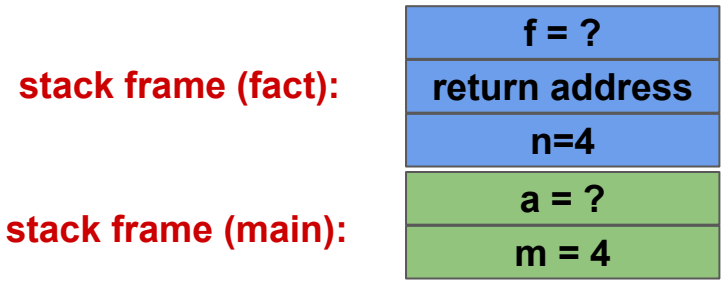
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

stack frame (main):

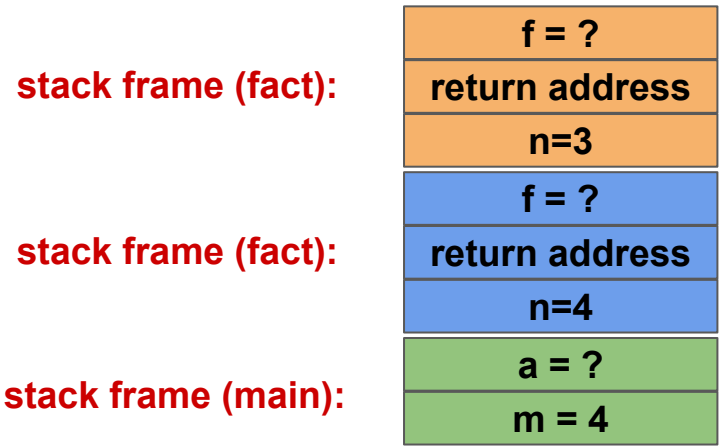


Call stack

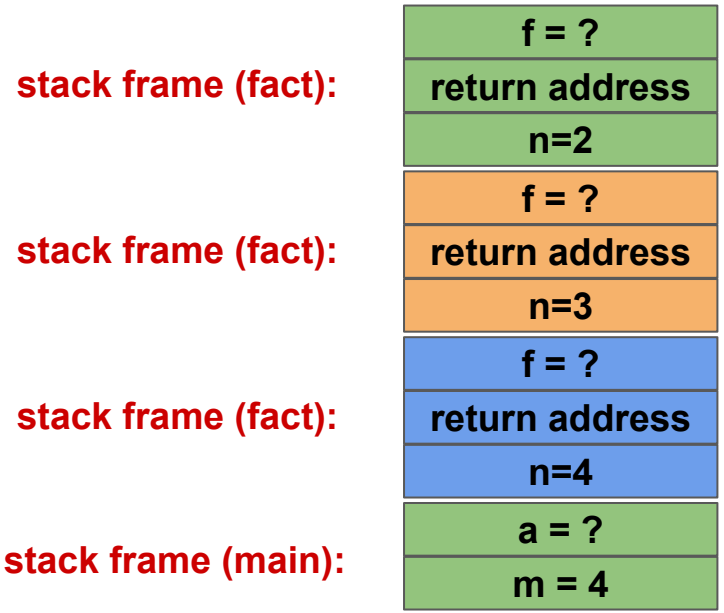
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



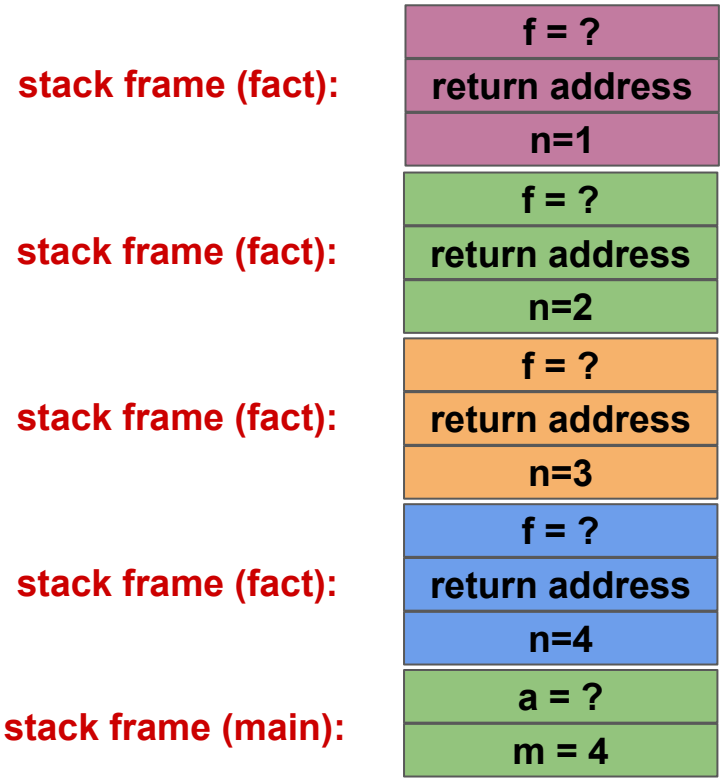
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

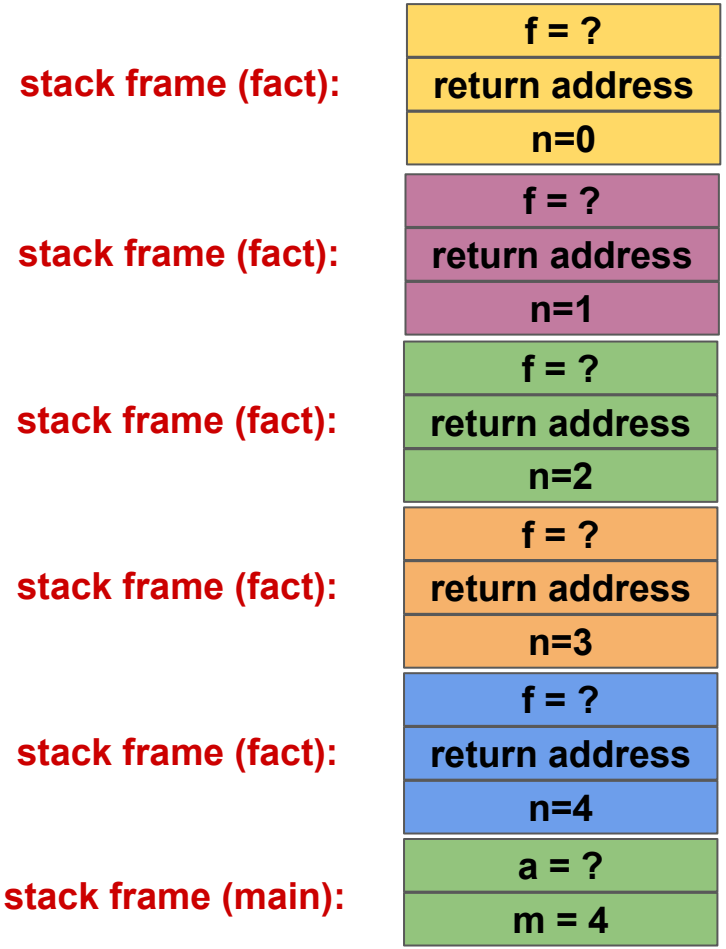


```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

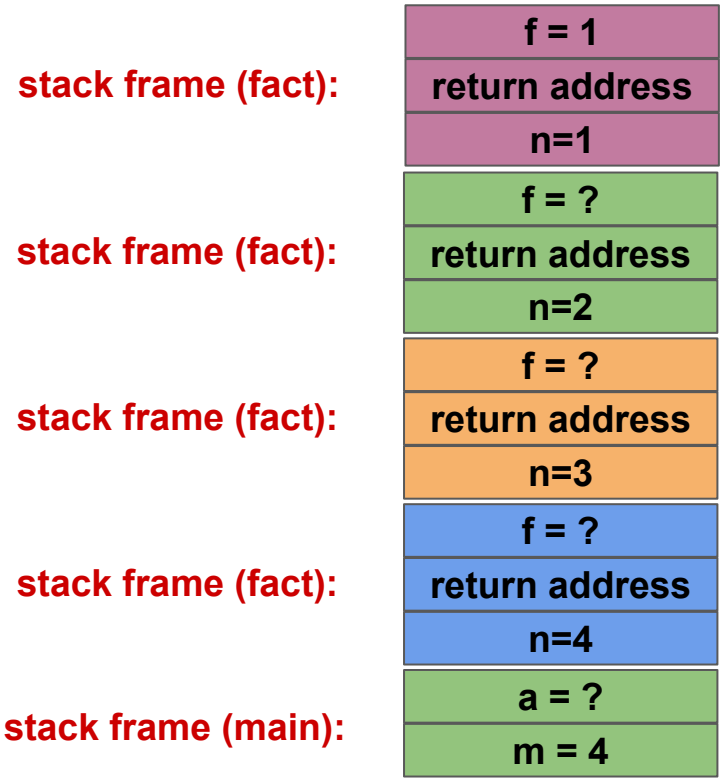


Call stack

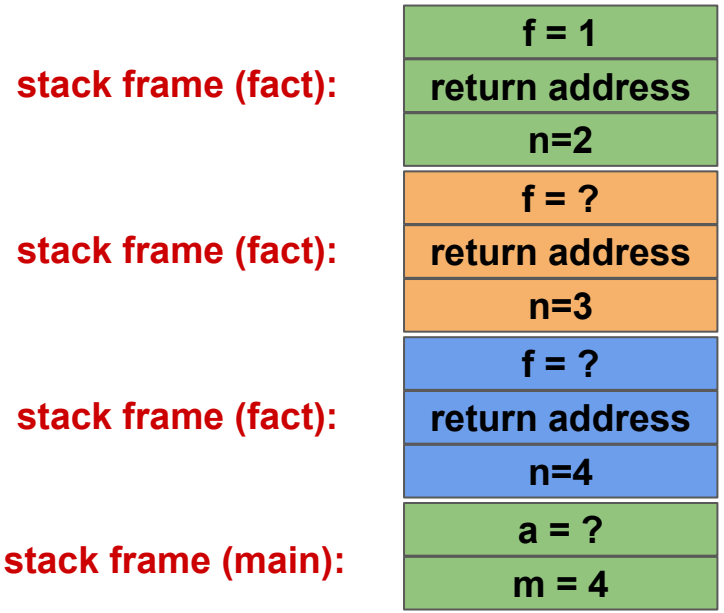
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



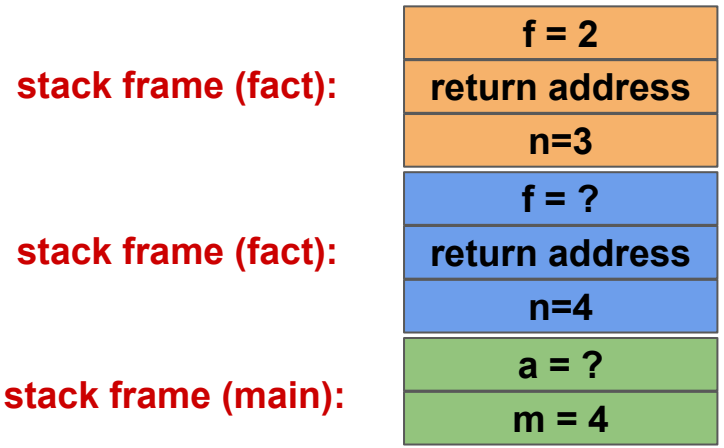
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



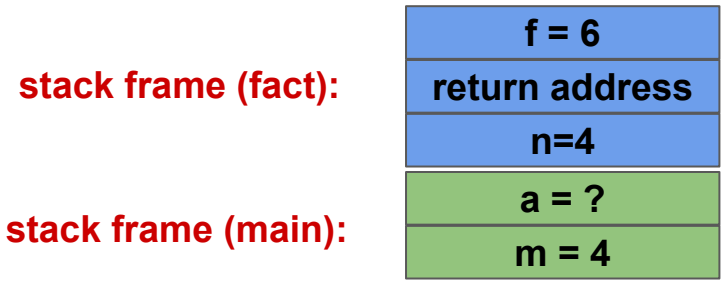
```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

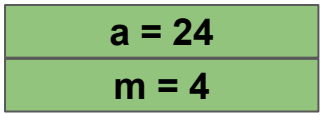


```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```



```c
#include <stdio.h>

int fact(int);


int main() {
  int m = 4;
  int a = fact(m);

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

}


int fact(int n) {
  double f;

  if (n == 0)
    return 1;

  f = fact(n-1);

  return n*f;

}
```

stack frame (main):



Recursion

Fibonacci series

- `f(1) = 1`
- `f(2) = 1`
- `f(n) = f(n) + f(n-1)`