

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

- 1 Recursive Mathematical Formulas
 - the factorial of a natural number
 - tracing a recursive execution
 - an accumulating parameter

Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

- 2 Recursion on STL Lists
 - generating n random numbers
 - writing a list recursively
 - searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

- 3 Recursive Greatest Common Divisor
 - computing the greatest common divisor recursively

MCS 360 Lecture 21
Introduction to Data Structures
Jan Vershelde, 11 October 2010

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number

tracing a recursive execution

an accumulating parameter

Recursion on STL Lists

generating n random numbers

writing a list recursively

searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

1 Recursive Mathematical Formulas

the factorial of a natural number

tracing a recursive execution

an accumulating parameter

2 Recursion on STL Lists

generating n random numbers

writing a list recursively

searching a list

3 Recursive Greatest Common Divisor

computing the greatest common divisor recursively

the factorial of n Recursive
Mathematical
Formulasthe factorial of a
natural numbertracing a recursive
executionan accumulating
parameterRecursion on
STL Listsgenerating n random
numberswriting a list
recursively

searching a list

Recursive
Greatest
Common
Divisorcomputing the
greatest common
divisor recursively

Given a natural number n , its factorial $n!$ is

$$n! = n \times (n - 1) \times \cdots \times 2 \times 1.$$

Interpretation: #choices of n items without repetition.

For example: how many 3-letter words with a, b, c?

abc, acb, bac, bca, cab, cba # : $3! = 6$.

What is $0!$? How many ways to choose nothing? $0! = 1$.

A recursive formula for $n!$ is

$$n! = \begin{cases} 1 & \text{if } n = 0, \\ n \times (n - 1)! & \text{if } n > 0. \end{cases}$$

the factorial of n Recursive
Mathematical
Formulasthe factorial of a
natural numbertracing a recursive
executionan accumulating
parameterRecursion on
STL Listsgenerating n random
numberswriting a list
recursively

searching a list

Recursive
Greatest
Common
Divisorcomputing the
greatest common
divisor recursively

Given a natural number n , its factorial $n!$ is

$$n! = n \times (n - 1) \times \cdots \times 2 \times 1.$$

Interpretation: #choices of n items without repetition.

For example: how many 3-letter words with a, b, c?

abc, acb, bac, bca, cab, cba # : $3! = 6$.

What is $0!$? How many ways to choose nothing? $0! = 1$.

A recursive formula for $n!$ is

$$n! = \begin{cases} 1 & \text{if } n = 0, \\ n \times (n - 1)! & \text{if } n > 0. \end{cases}$$

a recursive function

Recursive
Mathematical
Formulas

the factorial of a
natural number

tracing a recursive
execution

an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively

searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
int factorial(int n);  
// returns the factorial of n  
  
int factorial(int n)  
{  
    if(n==0)  
        return 1;  
    else  
        return n*factorial(n-1);  
}
```

stack of function calls

Recursive Mathematical Formulas

the factorial of a
natural number

tracing a recursive
execution

an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

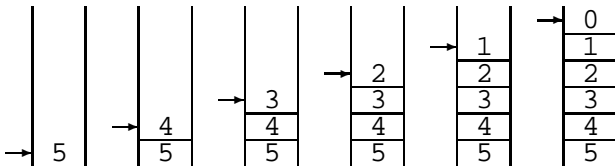
writing a list
recursively

searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Computing 5! recursively happens via a stack



- not in the base case: push argument on the stack
- after base case, pop from stack and evaluate

unwinding the recursion

Recursive
Mathematical
Formulas

the factorial of a
natural number

tracing a recursive
execution

an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively

searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```

factorial(5)
→ factorial(4)
  → factorial(3)
    → factorial(2)
      → factorial(1)
        → factorial(0)
          return 1
        return 1 * 1
      return 2 * 1 * 1
    return 3 * 2 * 1 * 1
  return 4 * 3 * 2 * 1 * 1
return 5 * 4 * 3 * 2 * 1 * 1

```

unwinding the recursion

Recursive
Mathematical
Formulas

the factorial of a
natural number

tracing a recursive
execution

an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively

searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
factorial(5)
→ factorial(4)
  → factorial(3)
    → factorial(2)
      → factorial(1)
        → factorial(0)
          return 1
            return 1 * 1
              return 2 * 1 * 1
                return 3 * 2 * 1 * 1
                  return 4 * 3 * 2 * 1 * 1
                    return 5 * 4 * 3 * 2 * 1 * 1
```

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number

tracing a recursive execution

an accumulating parameter

Recursion on STL Lists

generating n random numbers

writing a list recursively

searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

1 Recursive Mathematical Formulas

the factorial of a natural number

tracing a recursive execution

an accumulating parameter

2 Recursion on STL Lists

generating n random numbers

writing a list recursively

searching a list

3 Recursive Greatest Common Divisor

computing the greatest common divisor recursively

Recursive
Mathematical
Formulas

the factorial of a
natural number

tracing a recursive
execution

an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively

searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

tracing recursion

```
tracing 5! ...
    n = 5
    n = 4
    n = 3
    n = 2
    n = 1
n = 0
    returning 1
    returning 2
    returning 6
    returning 24
    returning 120
```

the function `trace_factorial`

Recursive Mathematical Formulas

the factorial of a
natural number
**tracing a recursive
execution**
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

```
int trace_factorial(int n)
{
    for(int i=0; i<n; i++) cout << " ";
    cout << "n = " << n << endl;

    if(n==0)
    {
        return 1;
        cout << "returning 1" << endl;
    }
    else
    {
        int r = n*trace_factorial(n-1);
        for(int i=0; i<n; i++) cout << " ";
        cout << "returning " << r << endl;
        return r;
    }
}
```

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

1 Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

2 Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

3 Recursive Greatest Common Divisor

computing the greatest common divisor recursively

an accumulating parameter

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution

**an accumulating
parameter**

Recursion on STL Lists

generating n random
numbers

writing a list
recursively

searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Instead of a tail recursion for $n!$,
we can accumulate the result in a parameter.

```
int accumulate_factorial(int n, int f)
{
    if(n <= 1)
        return f;
    else
        return accumulate_factorial(n-1, n*f);
}
```

tracing the execution

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution

an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
computing 5*1
computing 4*5
computing 3*20
computing 2*60
returning 120
```

```
int trace_accumulate_factorial(int n, int f)
{
    if(n <= 1)
    {
        cout << "returning " << f << endl;
        return f;
    }
    else
    {
        cout << "computing " << n << "*" << f << "\n";
        return trace_accumulate_factorial(n-1,n*f);
    }
}
```

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

- 1 Recursive Mathematical Formulas
the factorial of a natural number
tracing a recursive execution
an accumulating parameter

Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

- 2 Recursion on STL Lists
generating n random numbers
writing a list recursively
searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

- 3 Recursive Greatest Common Divisor
computing the greatest common divisor recursively

generating n random numbers

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

A recursive view of a nonempty list \mathbb{L} :
 \mathbb{L} has a node as head and a list as tail.

A recursive algorithm to generate n numbers:

- If n equals zero (or less) then
return an empty list;
- else (n is larger than zero)
 - generate a list \mathbb{L} of n-1 numbers;
 - push a random number to \mathbb{L} .

generating n random numbers

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

A recursive view of a nonempty list L :
 L has a node as head and a list as tail.

A recursive algorithm to generate n numbers:

- If n equals zero (or less) then
return an empty list;
- else (n is larger than zero)
 - generate a list L of n-1 numbers;
 - push a random number to L .

the function generate

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
list<int> generate ( int n )
{
    if(n <= 0)
    {
        list<int> L;
        return L;
    }
    else
    {
        int r = rand() % 1000;
        list<int> L = generate(n-1);
        L.push_front(r);
        return L;
    }
}
```

tracing the execution

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

Making `generate` verbose with print statements:

```
generating 3 random numbers ...  
generate with n = 3 ...  
generate with n = 2 ...  
generate with n = 1 ...  
generate with n = 0 ...  
returning empty list  
pushing 73 to front ...  
pushing 249 to front ...  
pushing 807 to front ...
```

What is the content of the list on return?

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

Recursion on STL Lists

generating n random numbers

writing a list recursively
searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

- 1 Recursive Mathematical Formulas
the factorial of a natural number
tracing a recursive execution
an accumulating parameter

- 2 Recursion on STL Lists
generating n random numbers
writing a list recursively
searching a list

- 3 Recursive Greatest Common Divisor
computing the greatest common divisor recursively

writing a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

writing a list recursively

searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

A recursive algorithm to write a list L :

- if list L is empty then
we do nothing;
- else (L is not empty)
 - pop first item i from L ;
 - write i ;
 - write L ; // we have popped i

writing a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

writing a list recursively

searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

A recursive algorithm to write a list L :

- if list L is empty then
we do nothing;
- else (L is not empty)
 - pop first item i from L ;
 - write i ;
 - write L ; // we have popped i

writing a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers

writing a list recursively

searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

A recursive algorithm to write a list L :

- if list L is empty then
we do nothing;
- else (L is not empty)
 - pop first item i from L ;
 - write i ;
 - write L ; // we have popped i

the function `write`Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers

**writing a list
recursively**

searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
void write ( list<int> L )
{
    if (!L.empty())
    {
        list<int> K = L;
        cout << " " << K.front();
        K.pop_front();
        write(K);
    }
}
```

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

- 1 Recursive Mathematical Formulas
the factorial of a natural number
tracing a recursive execution
an accumulating parameter

Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

- 2 Recursion on STL Lists
generating n random numbers
writing a list recursively
searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

- 3 Recursive Greatest Common Divisor
computing the greatest common divisor recursively

searching a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Does a number e belong to a list L ?

- if the list L is empty then
return false;
- else if front of L equals e then
return true;
- else
 - pop front element from list L ;
 - return belongs e to L ? // L is smaller

searching a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Does a number e belong to a list L ?

- if the list L is empty then
return false;
- else if front of L equals e then
return true;
- else
 - pop front element from list L ;
 - return belongs e to L ? // L is smaller

searching a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Does a number e belong to a list L ?

- if the list L is empty then
return false;
- else if front of L equals e then
return true;
- else
 - pop front element from list L ;
 - return belongs e to L ? // L is smaller

searching a list recursively

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Does a number e belong to a list L ?

- if the list L is empty then
return false;
- else if front of L equals e then
return true;
- else
 - pop front element from list L ;
 - return belongs e to L ? // L is smaller

the function belongs

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
bool belongs ( list<int> L, int e )
{
    if(L.empty())
        return false;
    else if(L.front() == e)
        return true;
    else
    {
        list<int> K = L;
        K.pop_front();
        return belongs(K,e);
    }
}
```

Recursive Definitions

Recursive Mathematical Formulas

the factorial of a natural number
tracing a recursive execution
an accumulating parameter

Recursion on STL Lists

generating n random numbers
writing a list recursively
searching a list

Recursive Greatest Common Divisor

computing the greatest common divisor recursively

- 1 Recursive Mathematical Formulas
the factorial of a natural number
tracing a recursive execution
an accumulating parameter

- 2 Recursion on STL Lists
generating n random numbers
writing a list recursively
searching a list

- 3 Recursive Greatest Common Divisor
computing the greatest common divisor recursively

the greatest common divisor

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Observe: $\text{gcd}(20,15) = \text{gcd}(15,5)$, $5 = 20 \% 15$.

A recursive algorithm to compute $\text{gcd}(a,b)$:

- base case: if $(a \% b == 0)$
then b divides a , so $b == \text{gcd}(a,b)$
- else, let $r = a \% b$, return $\text{gcd}(b,r)$.

Why is this an algorithm?

- termination: $r < b$; if $b > a$, then $r = a$
- correctness: $\text{gcd}(a,b) == \text{gcd}(b,a \% b)$

the greatest common divisor

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Observe: $\text{gcd}(20,15) = \text{gcd}(15,5)$, $5 = 20 \% 15$.

A recursive algorithm to compute $\text{gcd}(a,b)$:

- base case: if $(a \% b == 0)$
then b divides a , so $b == \text{gcd}(a,b)$
- else, let $r = a \% b$, return $\text{gcd}(b,r)$.

Why is this an algorithm?

- termination: $r < b$; if $b > a$, then $r = a$
- correctness: $\text{gcd}(a,b) == \text{gcd}(b,a \% b)$

the function gcd

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
int gcd(int a, int b)
{
    int r = a % b;
    if(r == 0)
        return b;
    else
        return gcd(b,r);
}
```

tracing the execution

Recursive
Mathematical
Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on
STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive
Greatest
Common
Divisor

computing the
greatest common
divisor recursively

```
give x : 98212
```

```
give y : 44632
```

```
gcd(98212,44632) = 4
```

```
tracing gcd(98212,44632) :
```

```
a = 98212, b = 44632, r = 8948
```

```
a = 44632, b = 8948, r = 8840
```

```
a = 8948, b = 8840, r = 108
```

```
a = 8840, b = 108, r = 92
```

```
a = 108, b = 92, r = 16
```

```
a = 92, b = 16, r = 12
```

```
a = 16, b = 12, r = 4
```

```
a = 12, b = 4, r = 0
```

Summary + Assignments

Recursive Mathematical Formulas

the factorial of a
natural number
tracing a recursive
execution
an accumulating
parameter

Recursion on STL Lists

generating n random
numbers
writing a list
recursively
searching a list

Recursive Greatest Common Divisor

computing the
greatest common
divisor recursively

Started Chapter 7 on recursive algorithms.

Assignments:

- 1 Draw the evolution of the stack of function calls for $5!$ computed recursively with an accumulating parameter.
- 2 Write a recursive function to sum a STL list of integer numbers. Give code to show that your function works.
- 3 Test what happens when the arguments for the gcd function would be negative numbers.

Third homework collection on Friday 15 October, at noon:
#1 of L-10, #2 of L-11, #2 of L-12, #3 of L-13, # of L-14.