VE280 Recitation Class (3)

Prepared by,
Zhang Yuhang
Yu JinZe
Cheng Songzhe
Yan Xuebin

Data Structure Summer
June 6th, 2012
F-Building, 200

Outline

- About Project One
- gdb
- Recursion & Tail recursion

Project One

- Submission
  - source: .C & .h
  - Makefile
  - testcases & docs

- Determining Similarity
  - MOSS (for a Measure Of Software Similarity)
  - http://theory.stanford.edu/~aiken/publications/papers/sigmod03.pdf

Project One

- NO Late Submission or Email Attachment for the rest of the semester.

- Project One Scores
- 20 points of compilation
- 13 testcases * 4 points each = 52 points
- 28 points coding style and submission issue
- At most 20 points great testcases.

- No Makefile, wrong Makefile, P1, Package files
- 5% deduction.

- Grading Feedback every 3 days

GDB

- g++ -g rec.C –o p1 gdb ./p1

- (gdb)set args a 12
  (gdb)run (r)
  (gdb)break(b) + function or (file:)line
  (gdb)list (l)
  (gdb)print(p)
  (gdb)watch
  (gdb)quit(q)
  (gdb)continue
  (gdb)next
  (gdb)step
  (gdb)delete

How to Debug Using GDB

- Debugging a program with a logical error
- Debugging a program that produces a core dump (a.k.a. Segmentation Fault)

Debugging a program with a logical error

- The program is supposed to output the summation of

  (X^0)/0! + (X^1)/1! + (X^2)/2! + (X^3)/3! + (X^4)/4! + ... + (X^n)/n!

- Given x and n as inputs.

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int ComputeFactorial(int number) {
  int fact = 0;

  for (int j = 1; j <= number; j++) {
    fact = fact * j;
  }

  return fact;
}

double ComputeSeriesValue(double x, int n) {
  double seriesValue = 0.0;
  double xpow = 1;

  for (int k = 0; k <= n; k++) {
    seriesValue += xpow / ComputeFactorial(k);
    xpow = xpow * x;
  }

  return seriesValue;
}
```

```cpp
int main() {
  cout << "This program is used to compute the value of the following series : " << endl;

  cout << "(x^0)/0! + (x^1)/1! + (x^2)/2! + (x^3)/3! + (x^4)/4! + ..... + (x^n)/n! " << endl;

  cout << "Please enter the value of x : " ;

  double x;
  cin >> x;

  int n;
  cout << endl << "Please enter an integer value for n : " ;
  cin >> n;
  cout << endl;

  double seriesValue = ComputeSeriesValue(x, n);
  cout << "The value of the series for the values entered is "
    << seriesValue << endl;

  return 0;
}
```

Debugging a program that produces a Segmentation Fault

- ulimit –c unlimited
  - to allow the system to generate core file when program crashes
- gdb [executable name] core

Debugging a program that produces a Segmentation Fault

```
1  #include <iostream>
2  void main()
3  {
4      char *temp = "Paras";
5      int i;
6      i=0;
7      temp[3]='F';
8      for (i =0 ; i < 5 ; i++ ) {
9          cout << temp[i] << endl;
10     }
11 }
```

Recursion

- 从前有座山，山上有座庙，庙里有个老和尚在给小和尚讲故事，他说从前有座山，山上有座庙，庙里有个老和尚在给小和尚讲故事，他说……
- procedure tell-story;
  begin
  if 讲话被打断 then 故事结束
  else begin
  从前有座山，山上有座庙，庙里有个老和尚在给小和尚讲故事，他说
  tell-story
  end
  end

Recursion

- 输入一个字符串，显示出串中所有字母可能的组合方式，用递归实现。
- 例如，输入abc
- 显示，abc,acb,bac,bca,cab,cba

Iteration VS Recursion

- Base case (stopping case)
  End condition

- Efficiency (e.g. factorial(n) in iteration and recursion?)
  Time complexity: time command
  space complexity

- Iteration <==> recursion

Tail Recursion

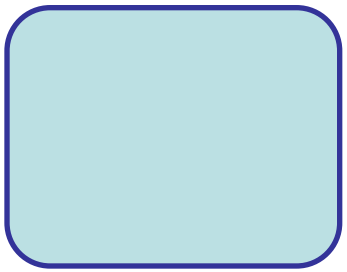
- Helper parameter

  “Recursive invariant” of recursion_helper

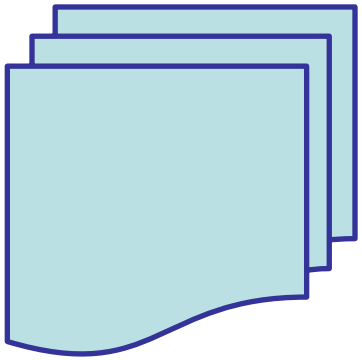
- Call stack

  With tail recursion, there is no pending computation at each recursive step, so we can **re-use** the activation record rather than create a new one.

Tail Recursion



Recursion



Fibonacci

Recursion: O(1.618^n)

```
int Fib(int n) {
    if(n==1)      return 1;
    if(n==2)      return 1;
    else          return fib(n-1)+fib(n-2) ;
}
```

Iteration: O(n)

```
Int Fib(int n)
{
    int a, b; a = 1; b = 0;
    for(int i=1;i<=n-1;i++)
    {
        sum=a+b;   b=a;   a=sum;
    }
    return sum;
}
```

Tail recursion: O(n)

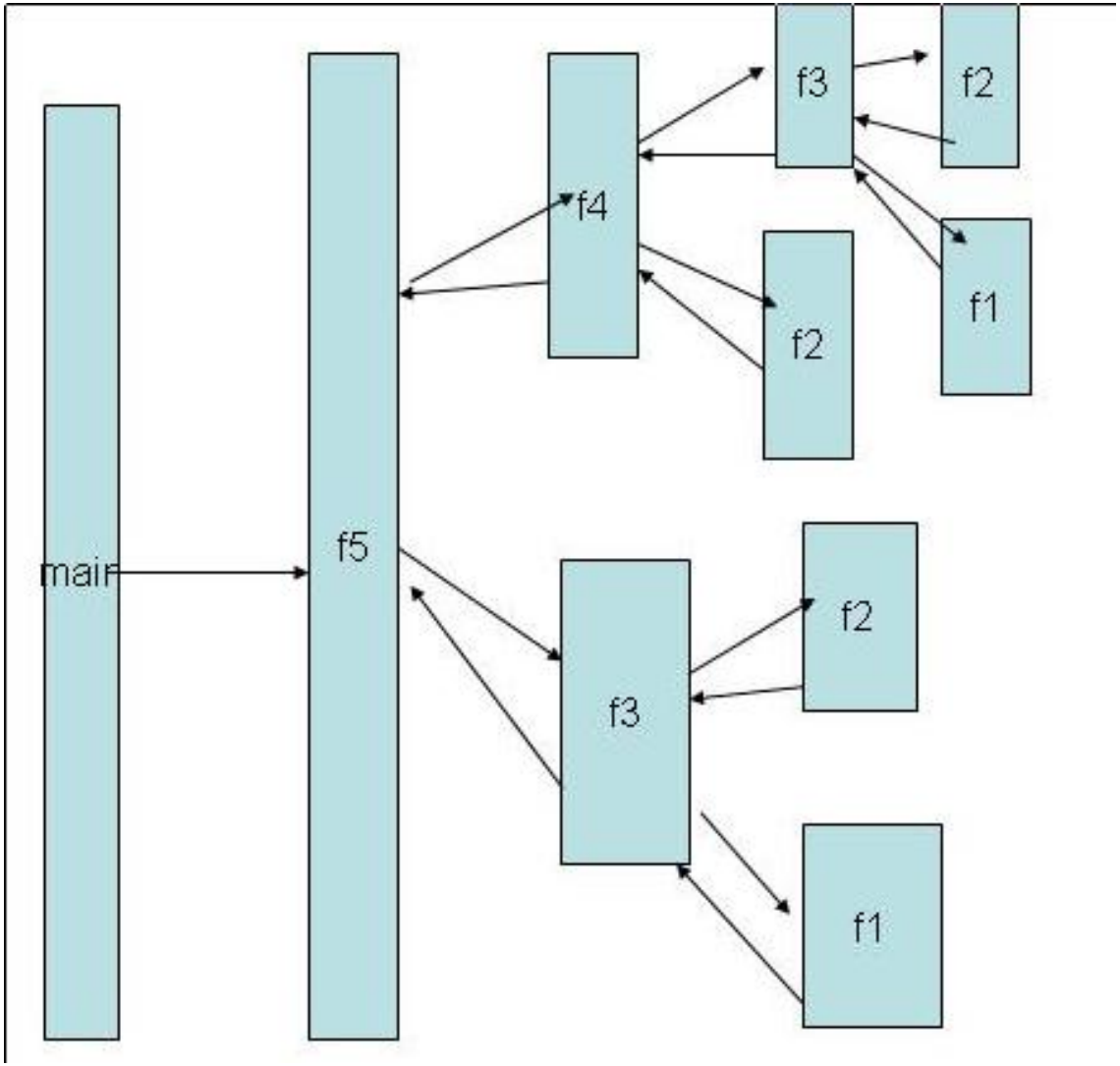
```
int Fib(int n)
{
   if (n == 1 || n == 2)
      return 1;
   else
      return Fib_tailhelper(n, 1, 1, 3);
}
int Fib_tailhelper(int n, long b1, long b2, int
begin)
{
   if (n == begin)
      return b1 + b2;
   else
      return Tailhelper(n, b2, b1 + b2, begin + 1);
}
```

```
zhangyuhang@ubuntu:~/VE280/RA$ ./run 46
rec:
1836311903

real    0m34.645s
user    0m34.082s
sys     0m0.032s
---------------
tail:
1836311903

real    0m25.755s
user    0m25.406s
sys     0m0.028s
---------------
iter
1836311903

real    0m0.009s
user    0m0.000s
sys     0m0.004s
---------------
```

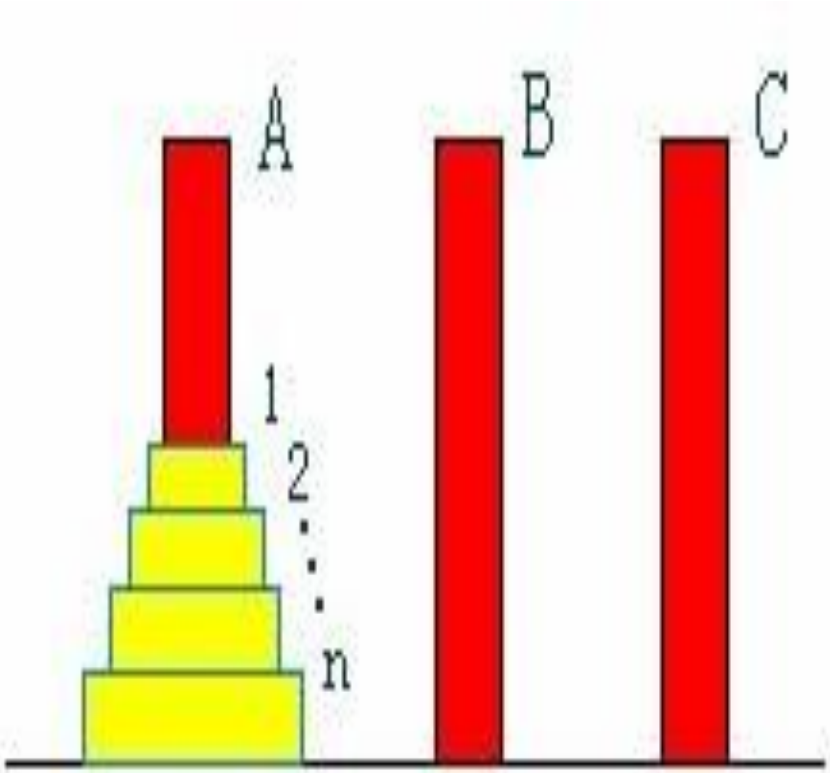
Fibonacci Stack Frame

Recursion:

```
int Fib(int n)
{
    if(n==1)
        return 1;
    if(n==2)
        return 1;
    else
        return fib(n-1)+fib(n-2) ;
}
```



Hanoi



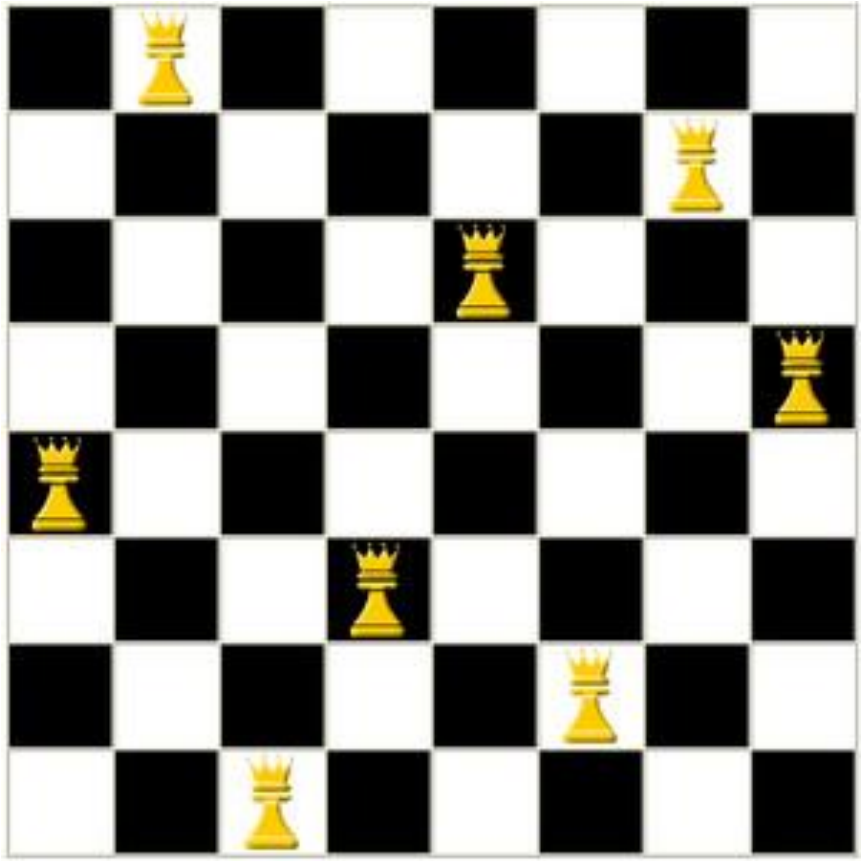
(D: Deck, P: Pad)

```
Hanoi(A, B, C, D(n))
{
    if (n == 1)
        move(A,C,P(1));
    else {
        Hanoi(A,C,B, D(n-1));
        move(A,C, P(n));
        Hanoi(B, A,C, D(n-1));
    }
}
```

Eight Queens

3~5 points bonus for students under 40/72 points

No two queens share the same row, column, or diagonal



No source code from the Internet.

Email to any of us TAs with your name and student ID.