Outline

• Number representation
  – Floating-point number
  – Two’s complement
• Recursive program
  – Introduction
  – Simple example
  – Compute GCD of two numbers

Question

How would the number –45.1875 be represented as a 32 bit single precision number according to the IEEE Floating Point Standard?

Answer

• 1) sign “1”
• 2) 45 = 101101
• 0.1875 = 0.0011
• 3) 45.1875 = 1.011010011 * 2^5
• 4) Exponent = 127 + 5 = 132 = 10000100
• 5) 1100001000110001100000….00000

Question

• What is the value of the following, 0x8FA40000
If it is
• 1) a 2’s complement integer
• 2) a floating point number
• 3) a MIPS instruction
• 4) 4 characters

Solution

1) 1000 1111 1010 0100 0000 0000 0000 0000
   -1: 1000 1111 1010 0011 1111 1111 1111 1111
   absolute value:
   0111 0000 0101 1100 0000 0000 0000 0000
   value = (-2^30 + 2^29 + 2^28 + 2^22 + 2^20 + 2^19 + 2^18)
2) Negative, exponent (00011111-127) = -96
   Value = -1.25 * 2^(-96)
2) lw $4,0($29) or lw $a0, 0($sp)
3) Å,ß,NULL,_NULL

Recursive Program

Recursion occurs when a function calls itself directly or indirectly.
• Base case (when the program terminates)
• Recursive function
  – Invariance, relations between sub-problems
Step: Check if base case is satisfied
  if NO, divide into sub program and call sub program
  if YES, merge the results and return
A simple example

• Assume that the function of adding two numbers together were not provided in the program instruction set. You can only use the operator ++, and operator --. Can you write a recursive pseudo program for function add(i,j) which return the sum of i and j?

Solution

• Base case
  i = 0, add(i,j) = j
  Or j=0, add(i,j) = i
• Recursive function
  add(i,j) = add(--i,++j)
• Pseudo codes
  add(i,j)
  – if (i==0) return j
  – else return add(--i,++j)

Convert it into MIPS

Any problem with the following code?
Answer: $ra should be saved, otherwise the program loops forever between move $v0,$a1 and jr $ra

#parameters in $a0,$a1
#return value in $v0
Add_two:  beq $a0,$zero,Return
         subi $a0,$a0,1
         addi $a1,$a1,1
         jal Add_two
Return:  move $v0,$a1
         jr $ra

GCD of M and N

• GCD greatest common divisor (GCD) of two integers m and n, which is the greatest integer that divides both m and n with no remainder.
• //m>=n>0
  int gcd(int m, int n) {
    if ((m % n) == 0)    return n;
    else return gcd(n, m % n);}