CS19001: Programming and Data Structures Laboratory

Soumyajit Dey, Aritra Hazra;
CSE, IIT Kharagpur

http://cse.iitkgp.ac.in/~aritrah/course/lab/PDS/Autumn2018/CS19101_PDS-Lab_Autumn2018.html

20-Aug-2018

Iterative execution (Loops)

The while loop

```
while (condition)
{
   execute loop body;
}
```

GCD by repeated division

```
while (b > 0)
{
   r = a % b;
   a = b;
   b = r;
}
printf("gcd = %d\n",a);
```

Iterative execution (Loops)

The for loop

```
for ( initialize; condition; increment )
{
   execute loop body;
}
```

N^{th} harmonic number $H(n) = \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n}$

```
H = 0;
for (i=1; i<=n; ++i) H += 1.0/i;
   printf("H(%d) = %f\n", n, H);
```

Iterative execution (Loops)

The Fibonacci numbers

$F_n = F_{n-1} + F_{n-2}$ for $n \geq 2, F_1 = 1, F_0 = 0$

While

```
i = 1, F = 1; p1 = 0;
while (i < n)
{
  ++i;
  p2 = p1;
  p1 = F;
  F = p1 + p2;
}
```

For

```
p1 = 0, F = 1;
for(i = 2;i <= n;++i)
{
  p2 = p1;
  p1 = F;
  F = p1 + p2;
}
```

```
printf("F(%d)=%d",n,F); //for both programs
```

Loop flow control

- A loop may be *conditionally* broken from inside

```
while (1)
{
 if (b == 0) break;
 r = a % b;
 a = b;
 b = r;
}
printf("gcd = %d\n", a);
```

Loop flow control

- A loop iteration may be *conditionally* skipped
- Ex: Printing 1,2,...,100 neatly with 10 integers per line

```
for (i=1; i<=100; ++i) {
    printf("%4d",i);
    if (i%10 != 0) continue;
    printf("\n");
}
```

Debugging you program: removing logical errors

- First look at your program and see if you can find some obvious logical errors. If found, correct and retry
- If it is not immediately evident, take some (small) input, work out by hand what the values of your variables should be after each step logically
- Put printf statements at those steps and find the first step the program prints a wrong value. Keep repeating until all mistakes are corrected

Bug Localization


```
void main()
{
 int k = 2, n = 1;
 while (k < 7) {
  n = n*k;
  k++;
 }
 printf("After loop 1\n"); /*printf for debugging*/
 while (k != 21) {
  n = n + k;
  k = k+2;
 }/* do not miss \n in debug printf */
 printf("After loop 2\n"); /*printf for debugging*/
 printf("n is %d\n", n);
}
```

- Program hangs, second loop does not terminate
- Statemnent "After loop 2" is not printed, So you know the first loop finished and the second did not.

GCD: erroneous implementations

Correct

```
while (1)
{
 if (b == 0) break ;
 r = a % b;
 a = b;
 b = r;
}
```

always o/p = 0

```
while (1)
{
 if (b == 0) break ;
 r = a % b;
 b = r;
 a = b;
}
```

```
/*last 2 statements exchanged*/
```

o/p is in 'a'. In R.H.S program, a=0 due to the chaining effect when 'r' is 0

Debugging a single block

Executing the correct program with a=45, b=12

```
 while (1)
 {
  if (b==0) break ;
  r = a % b;                    /* iter 1 values*/
  printf("a=%d,b=%d,r=%d\n"); /* 45,12,9*/
  a = b;
  printf("a=%d,b=%d,r=%d\n"); /* 12,12,9*/
  b = r;
  printf("a=%d,b=%d,r=%d\n"); /* 12, 9,9*/
 }
printf ("gcd = %d\n", a);
```

We expect a = old value of b = 12 , b = r = a%b =9
so, this is fine

Debugging a single block

Executing the incorrect program with a=45, b=12

```
 while (1)
 {
  if (b==0) break ;
  r = a % b;                    /* iter 1 values*/
  printf("a=%d,b=%d,r=%d\n"); /* 45,12,9*/
  b = r;
  printf("a=%d,b=%d,r=%d\n"); /* 45, 9,9*/
  a = b;
  printf("a=%d,b=%d,r=%d\n"); /*  9, 9,9*/
 }
printf ("gcd = %d\n", a);
```

We expect a = old value of b = 12 , b = r = a%b =9

Only r is assigned correctly, problem with code after r=a%b

GCD: some more erroneous implementations :)

Infinite loop

```
while (1)
{
 if (b == 0) break ;
 r = a % b;
 a = b;
 b = a;
}
/*b=a by mistake*/
```

Divide by zero

```
 while (1)
{
 if (a == 0) break ;
 r = a % b;
 a = b;
 b = r;
}
/*a==0 by mistake*/
```

Nested loop:

```
int i, j;
/* print header line: */
printf("   ");
for(j = 1; j <= 10; j = j + 1)
  printf(" %3d", j);
printf("\n");
/* print table: */
for(i = 1; i <= 10; i = i + 1)
{
  printf("%2d", i);
  for(j = 1; j <= 10; j = j + 1)
    printf(" %3d", i + j);
  printf("\n");
}
return 0;
```

Output table

	1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10	11
2	3	4	5	6	7	8	9	10	11	12
3	4	5	6	7	8	9	10	11	12	13
4	5	6	7	8	9	10	11	12	13	14
5	6	7	8	9	10	11	12	13	14	15
6	7	8	9	10	11	12	13	14	15	16
7	8	9	10	11	12	13	14	15	16	17
8	9	10	11	12	13	14	15	16	17	18
9	10	11	12	13	14	15	16	17	18	19
10	11	12	13	14	15	16	17	18	19	20

Make a simple modification to the program to print a multiplication table, or a subtraction table

break statement in loop nest

```
void main()
{
   int low, high, desired, i, flag = 0;
   scanf("%d %d %d", &low, &high,&desired);
   i = low;
   while (i < high) {
      for (j = i+1; j <= high; ++j) {
          if (j % i == desired) {
              flag = 1;
              break; //breaks from for loop
          }
      }
      if (flag == 1) break;
      i = i + 1; //breaks from while loop
   }
}
```

Soumyajit Dey, Aritra Hazra; CSE, IIT Kharagpur

Programming Assignments

Complete and submit during lab

Assignment 1

Soumyajit Dey, Aritra Hazra; CSE, IIT Kharagpur

Write a program which will take as input an integer x and print out the value of $x!$, factorial of x. We will like to compute factorial of reasonably large integers, say 10, 11, etc.

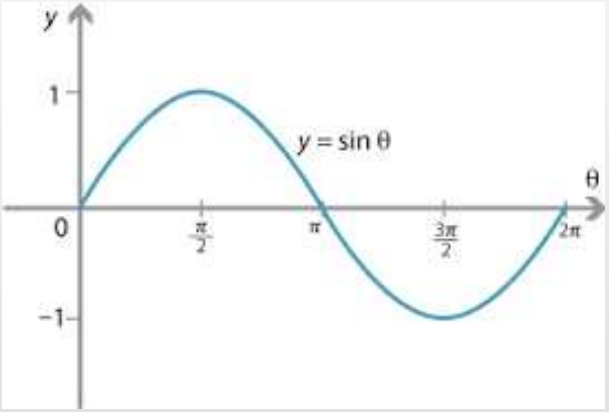
However, say 15! cannot be stored in a 32-bit integer data type, so for computing factorial, use "double" as a data type and avoid errors.

Assignment 2

Write a program which performs the following.

- requests the user for an integer i.
- prints out the value of sin function inside the interval $[0, 2\pi]$ at $i + 1$ uniformly placed points.
- use float as a data type for input and double as a data type for output of sin function (math.h).
- Use "M_PI", a constant defined in math.h as the value of π. You can directly use it in your code as it is already defined in the header file.

Assignment 2: Expected Output




```
Enter resolution:  6
sin(0.00 pi)= 0.000000
sin(0.33 pi)= 0.866025
sin(0.67 pi)= 0.866025
sin(1.00 pi)= -0.000000
sin(1.33 pi)= -0.866025
sin(1.67 pi)= -0.866025
sin(2.00 pi)= 0.000000
```

Assignment 3

Ask the user to provide a resolution r which is any float value (preferably smaller than 0.0004). Modify the previous program as follows.

- Stop using the math.h sin function. Note that $sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots$
- Implement a loop for computing the above series where in each iteration one term is +/-
- Break out of loop when the difference between the partial sums computed in the last two iterations is $< r$.
- suppose, $a = x - \frac{x^3}{3!} + \frac{x^5}{5!}, b = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!}$ and $|a - b| \leq r$, then break from loop and report $[a, b]$ as approximate value of sin(x) if $a \leq b$ or $[b, a]$ if $b < a$.

Assignment 3 : Useful Hints

- For safe handling of precision, use “double” as data type.
- Use fabs() function from math.h for computing modulus.

Assignment 3: Expected Output

```
Enter resolution:  6
Enter Interval size:  0.0004
sin(0.00 pi)= 0.000000
my sin(0.00 pi)= [0.000000, 0.000000]
sin(0.33 pi)= 0.866025
my sin(0.33 pi)= [0.866021, 0.866295]
sin(0.67 pi)= 0.866025
my sin(0.67 pi)= [0.866023, 0.866108]
sin(1.00 pi)= -0.000000
my sin(1.00 pi)= [-0.000001, 0.000021]
sin(1.33 pi)= -0.866025
my sin(1.33 pi)= [-0.866126, -0.866020]
sin(1.67 pi)= -0.866025
my sin(1.67 pi)= [-0.866049, -0.865672]
sin(2.00 pi)= 0.000000
my sin(2.00 pi)= [-0.000005, 0.000083]
```

Thank You