Tutorial

Programming & Data Structure: CS 11001

Section - 4/D

Department of Computer Science and Engineering

I.I.T. Kharagpur

Spring Semester: 2013 - 2014 (27.03.2014)

Download

Download the file tut270314.pdf from Programming & Data Structures ... of

http://cse.iitkgp.ac.in/~goutam

View the file using the command acroread & or xpdf &

Tutorial XI.1

Write a C program that will dynamically allocate a 2D-array like structure with r rows and 10 columns ($r \times 10$) using the pointer variable `int (*q)[10]`, where `r` is an input.

Tutorial XI.2

After the memory allocation,

1. Where does `q` point to?
2. Where does `q+i` point to?
3. What is `*q` ?
4. What is `*q+j`?
5. What is `*(q+i)+j`?
6. Explain `**q`, `*(*q+j)`, `*(*(q+i)+j)`.

Tutorial XI.3

Write a C program that will dynamically allocate a 2D-array like structure of 10 rows and c columns ($10 \times c$), using the array of pointers `int *r[10]`, where `c` is an input.

Tutorial XI.4

After the memory allocation,

1. Where does `r` point to?
2. Where does `r+i` point to?
3. What is `*r` ?
4. What is `*r+j`?
5. What is `*(r+i)+j`?
6. Explain `**r`, `*(*r+j)`, `*(*(r+i)+j)`.

Tutorial XI.5

Write a C program that will dynamically allocate a 2D-array like structure with r rows and c columns ($r \times c$), using the pointer variable `int **s`, where `r` and `c` are two input.

Tutorial XI.6

After the memory allocation,

1. Where does `s` points to?
2. Where does `s+i` points to?
3. What is `*s` ?
4. What is `*s+j`?
5. What is `*(s+i)+j`?
6. Explain `**s`, `*(*s+j)`, `*(*(s+i)+j)`.

Pseudo-random Number

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
int main()              // randPi.c
{
    double x, y, randMax ;
    int n, i, inC=0, inS=0;
```

```
srand(getpid());
printf("Enter data count: ");
scanf("%d", &n);
randMax = (double)RAND_MAX;
for(i=1; i<=n; ++i){
    x = rand()/randMax;
    y = rand()/randMax;
    inS++;
    if(x*x+y*y <= 1.0) inC++;
}
printf("Ratio: %f\n", 4.0*inC/inS);
```

```
    return 0;
}
```

Tutorial XI.7

What is the expected output of the code when n is large?

Quick Sort Algorithm

```
quickSort(a, l, h)
if (l < h)
    p ← partition(a, l, h)
    a[l] ↔ a[p]
    quicksort(a, l, p-1)
    quicksort(a, p+1, h)
```

Partition Algorithm

```
partition(a, l, h)
p ← a[l]
i ← l
j ← h + 1
while TRUE do
    do i ← i + 1 while i < h + 1 and a[i] < p
    do j ← j − 1 while a[j] > p
    if (i < j) a[i] ↔ a[j]
    else return j
endWhile
```

Function Quicksort

```
void quickSort(int data[], int low, int high) {
     if(low < high) {
        int partIndex, temp ;
        partIndex=partition(data,low,high);
        EXCH(data[low], data[partIndex], temp);
        quickSort(data, low, partIndex-1) ;
        quickSort(data, partIndex+1, high) ;
     }
}
```

Tutorial XI.8

1. How do you modify the code to print the depth of recursion for `quickSort()`?
2. What are the depths of recursion for the following two data sets: `90 80 70 60 50 40 30 20 10` and `10 20 30 40 50 60 70 80 90`.