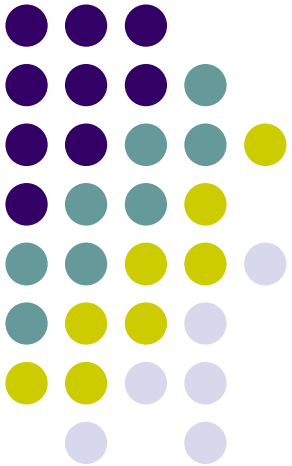
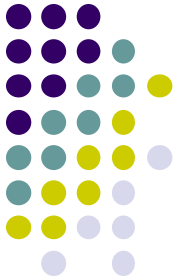
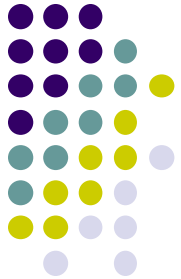
PDS Lab Section 15

January 21, 2021





General Instructions

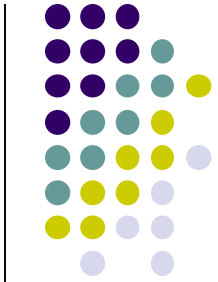


- Add header in front of your program

```
/*
*     Section 15
*     Roll No  :  20CS30010
*     Name     :  Your Name
*     Assignment No : 3
*     Description : Program to check points
*/
```

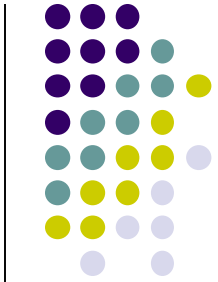
- Name your file with assgnX_Y.c, where X is the assignment number and Y is your roll no.
  - assgn9_20ME10010, assgn10_20CE30014,…..

Indenting your program



- Give proper indentation (space in front of a line) in your program to make your program look nice
- Indent every block in your program consistently
  - What is a block? – statements inside if, else, for, while etc…(for now)
  - Idea is to be able to see which statements are part of which if or which else or which for/while etc. easily
    - Helps in making less mistakes in missing braces etc. when you write programs
    - Helps in easier debugging if your program does not work

Badly indented program

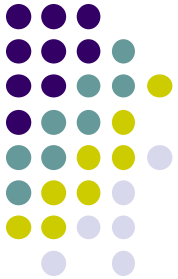


```
int main()
{
int i, j,k,n =10,sum, count;
for (i=0; i<n; i++)
{
sum=0; count=0;
for (j=0;j<n;j++)
{ scanf("%d",&k);
if (k>0)
{sum = sum + k;
 count = count + 1;
}
printf("Sum=%d Count=%d\n", sum, count);
}
}
}
```

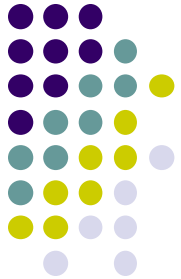
Properly Indented Program

```
int main()
{
    int i, j,k,n = 10, sum, count;
    for (i=0; i<n; i++)
    {
        sum=0;
        count = 0;
        for (j=0; j<n; j++)
        {
            scanf("%d",&k);
            if (k>0)
            {
                sum = sum + k;
                count = count + 1;
            }
            printf("Sum=%d Count=%d\n", sum, count);
        }
    }
}
```

Commenting a function



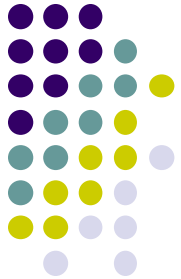
- Before each function declaration, add a comment with the following
  - The name of the function
  - A one line description for each parameter
  - A one line description of the return value
  - A small description of what the function does



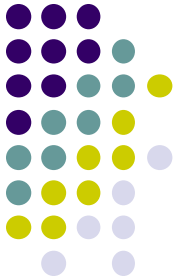
- Example:

```
/*****************
Function name: FindMin
Parameters:
        A : array of integers
        n : no. of integers in the array A
Return value: The minimum integer in array A
Description: This function finds the minimum value of a set of
             integers stored in array A
*****************/

int FindMin(int A[ ], int n)
{
   ….
}
```

Intermediate Submission for the First Assignment

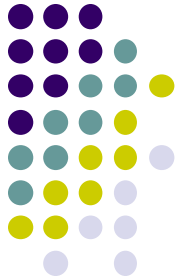


- For the first assignment, you will see two submission links in moodle
  - Assignment 11 Submission (Intermediate)
  - Assignment 11 Submission (Final)
- You **must submit** the .c file containing whatever you have done so far for Assignment 11 (first assignment today) in the link "Assignment 11 Submission (Intermediate)" strictly between 10-30 am to 10-45 am (**the link will open at 10-30 am and close at 10-45 am**)
  - Submit whatever you have done till then, we want to see how regularly you are progressing
  - **Not submitting will incur 30% penalty irrespective of what your final submitted version is**
- Submit the final .c file using the link "Assignment 11 Submission (Final)" as usual anytime before the lab ends (**must submit for grading, only the final version will be graded**)



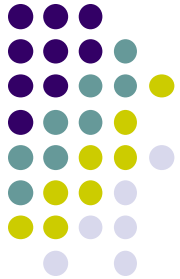
Today's Topic: Recursive Functions and Structures

Recursive Functions



- Functions that call itself
- Revise functions very well
- Any recursive function must have
  - A call to the same function
  - A terminating/base condition to indicate when to stp making recursive call
  - When you write a recursive function
    - Fist write the terminating/base condition
    - Then write the rest of the function
    - Always double-check that you have both

Example: Finding max in an array

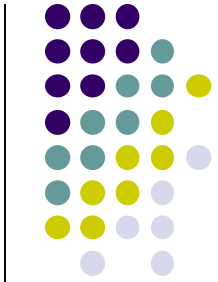


```
int findMax(int A[ ], int n)
{
        int temp;
        if (n==1)
        {
                return A[0];
        }
        temp = findMax(A, n-1);
        if (A[n-1] > temp)
                return A[n-1];
        else return temp;
}
```

Terminating condition. Small size problem that you know how to solve directly without calling any functions

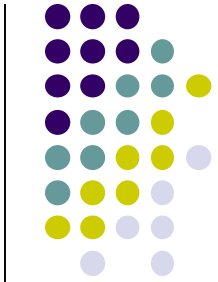
Recursive call. Find the max in the first n-1 elements (exact same problem, just solved on a smaller array).

Important things to remember



- Think how the whole problem (finding max of n elements in A) can be solved if you can solve the exact same problem on a smaller problem (finding max of first n-1 elements of the array). But then, do NOT think how the smaller problem will be solved, just call the function recursively and assume it will be solved
- For most problems you will see in this course, you will not need to use both loops and recursive calls in a recursive function. So if you have loops inside your recursive function, think whether you really need it.

Structures

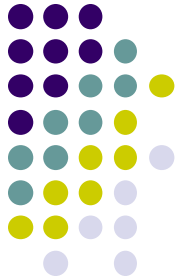


- Composite data type
  - Holds more than one data (more than one “field”)
  - Used to group related data
  - Example: a structure for keeping all information about a student like name (string), roll no. (string), age (integer), address (string), CGPA (float),…

```
struct student {
    char name[50];
    char roll_no[10];
    int age;
    char address[200];
    float CGPA;
};
```

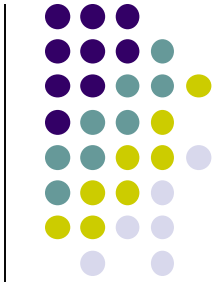
/* Very important to give the ; at the end of the } after the declaration */

Example of reading/writing structures

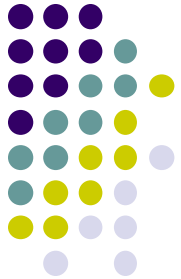


```
 /* Define a structure Point */
struct point {
   int x ; /* stores the x co-ordinate */
   int y ; /* stores y co-ordinate */
} ;

int main() {
   /* Declare struct Point type variable */
   struct point p;

  /*read in x and y co-ordinate for the point in p */
   scanf("%d%d",&p.x, &p.y) ;

  /* Now print the x and y co-ordinate of p */
  printf("\nYou have entered the following point : (%d,%d)", p.x, p.y) ;
}
```

Example: Finding distance between two points

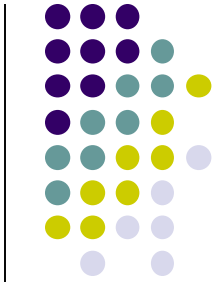


```
float CalculateDistance(struct point, struct point);

int main()
{
   /* Declare struct Point type variables */
   struct point p1,p2;
   float distance ;

   /* read x and y co-ordinates of the two points */
   printf(“\nEnter x, y co-ordinate of 1st point ");
   scanf(“%d%d”, &p1.x, &p1.y);
   printf("\nEnter x, y co-ordinate of 2nd point :");
   scanf(“%d%d”, &p1.x, &p1.y);

   /* Calculate the distance */
   distance = CalculateDistance(p1,p2) ;

   /* now print the calculated distance */
   printf("\nDist bet. the points are %d”, distance) ;
}
```



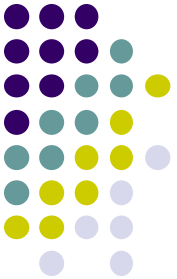
```
/* Function to calculate the distance between two points */

float CalculateDistance(struct point p1, struct point p2)
{
    int x_diff, y_diff ;
    double dist;
    x_diff = p1.x - p2.x ;
    y_diff = p1.y - p2.y ;
    dist = sqrt(pow(x_diff,2) + pow(y_diff,2)) ;
    return (dist);
}
```

Example: Reading/Writing Student records



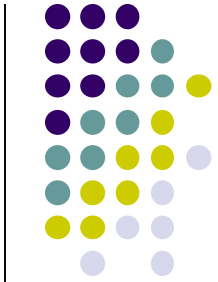
```
int main() {
    /* Declare an array of 100 students */
    struct student studentList[100] ;
    int number_of_students ;
    int idx ;

    /* scan the number of the students */
    printf("\nEnter the number of the student : ") ;
    scanf("%d",&number_of_students) ;

    /* Now read all the student information */
    for(idx = 0 ; idx < number_of_students ; idx++) {
        printf("\nEnter the name : ");
        scanf("%s",studentList[idx].name) ;
        printf("Enter the roll number : ");
        scanf("%s",studentList[idx].roll_no) ;
        printf("Enter the CGPA : ");
        scanf("%f",&studentList[idx].CGPA) ;
    }
```

Example: Reading/Writing Student records (contd.)



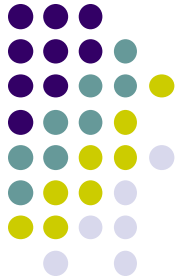
```
  /* Now print the student information */
  for(idx = 0 ; idx < number_of_students ; idx++) {
     printf("\nPrinting data for %dth student:\n",idx);
     printf("Name: %s\n",studentList[idx].name);
     printf("Roll No: %s\n", studentList[idx].roll_no);
     printf("CGPA    : %f\n",studentList[idx].CGPA);
     printf("\n") ;
  }

}
```

Practice Problem 1

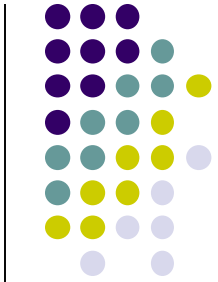


- Finding if two circles intersect
  - First create a struct named circle to represent the circle
    - May contain two int to represent the x and y coordinate of the center and a float to represent the radius of the circle
  - Now define two circles (two variables of type struct circle), and read in the details.
  - Declare a function to check whether the two circles intersects or not
    - int IsIntersectingCircle(struct circle, struct circle)
  - Two circles intersect if the distance between their centers is less than the sum of their radii.

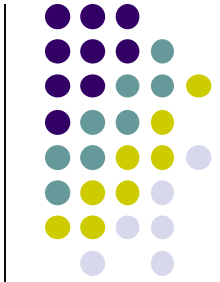
Practice Problem 2



- Searching for a roll number and printing corresponding CGPA
  - Create a structure for storing student record called student
    - May contain an int roll number field and a float CGPA field.
  - Declare an array to hold 100 student records
  - Read an integer n, n < 100, as the number of students
  - Read in the student details for all students in the array.

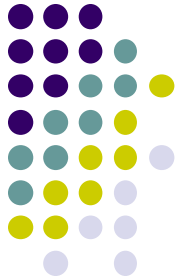


- Now, read another int as the roll number to be searched.
- Search the array of student records to match the roll number field
  - If the match occurs then print the corresponding CGPA fields of that record.
  - If no match occurs, print an appropriate message

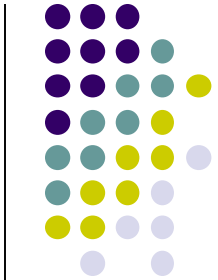


Assignments

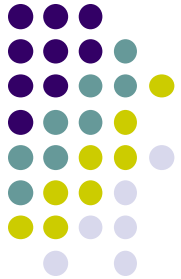
Assignment 11



- A line segment can be defined by its two endpoints
- A rectangle is said to be axis-parallel if its sides are parallel to the x and y-axis
- An axis parallel rectangle can be fully defined by two points – its top left corner point and bottom right corner point

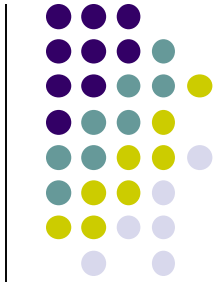


- Define a C structure called line to store a line segments
  - i.e., store the x-y coordinates of its two endpoints
- Define a C structure named rectangle to store an axis-parallel rectangle
  - i.e. store the x-y coordinates of its top left and bottom right corners
- Write a function
  - int Intersects(struct rectangle R, struct line L)
  - that takes an axis parallel rectangle R and a line segment L parallel to x or y axis, and find the number of points in which L cuts R



- Write a main function to do the following:
  - Define a variable of type struct rectangle
  - Define a variable of type struct line
  - Read in the x-y coordinates of the two corners of the rectangle
  - Read in the x-y coordinates of the two endpoints of the line segment
  - Call Intersects( ) to find the number of intersections between the rectangle and the line segment
  - Print the number of intersections

Assignment 12



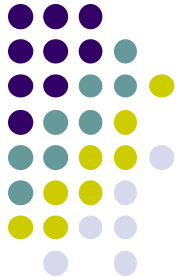
A string X of length n is said to be a prefix of a string Y of length m if m ≥ n and if the first n characters of y are the same as the characters of X. For example, the string “khar” is a prefix of the string “kharagpur” and the string “sun” is a prefix of the string “sun”.

Write a **recursive** C function

int IsPrefix(char X[ ], int l1, int h1, char Y[ ], int l2, int h2)

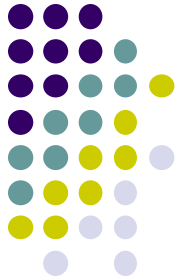
that returns 1 if the substring between index l1 and index h1 of X is a prefix of the substring between index l2 and h2 of Y. The function returns 0 otherwise.

For ex, if X = “screaming” and Y = “dreaming”, then IsPrefix(X, 2, 5, Y, 1, 6) will check if the substring “ream” in X (X[2] to X[5]) is a prefix of the substring “reamin” in Y (Y[1] to Y[6])



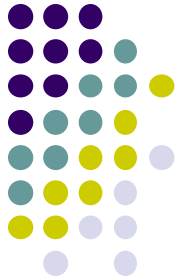
Write a main function that

- reads in two strings in two character arrays A and B using the %s option in scanf. You can assume that the strings will have at most 100 characters.
- finds the length of the two strings.
- calls the IsPrefix() function to find if the string in A is a prefix of the string in B, and print a message appropriately.
  - If the length of A is p1, and the length of B is p2, you will call IsPrefix() with the parameters (A, 0, p1 – 1, B, 0, p2 – 1).
- All printings must be done from main, there should not be anything printed inside the function.



- Hint:
  - First think of two complete strings and a recursive formulation to define when is one a prefix of another
    - So if X =“khar”, and Y = “kharagpur”, for X to be a prefix of Y, note that first letter of both has to be same (‘k’) and the remaining part of X (“har”) has to be a prefix of the remaining part of Y (“haragpur”). Do you see the recursion now?
  - Then just use the same idea, but on the substrings defined by the parameters
    - Decide what parameters to pass in the recursive call
    - Work with the above example to understand
  - For recursive calls made, store the return value in a variable. Then think what to do with it.

Teaser problem (not to be submitted)



- Let there be n coins of value denominations C = {c1, c2, ...cn}. A particular coin can be chosen only once. Given a required value V, you are to find the minimum number of coins from C whose total value exactly equals V. Write a recursive function for solving the problem. For example, if C={8,1,5,7,2,6}, V=11 the answer is {6,5}. In some examples there may not be a solution which also has to be pointed out in your function.