Ex 1. [Practice] Matrix Multiplication - Fixed

Github Commit date: 09-07-2023

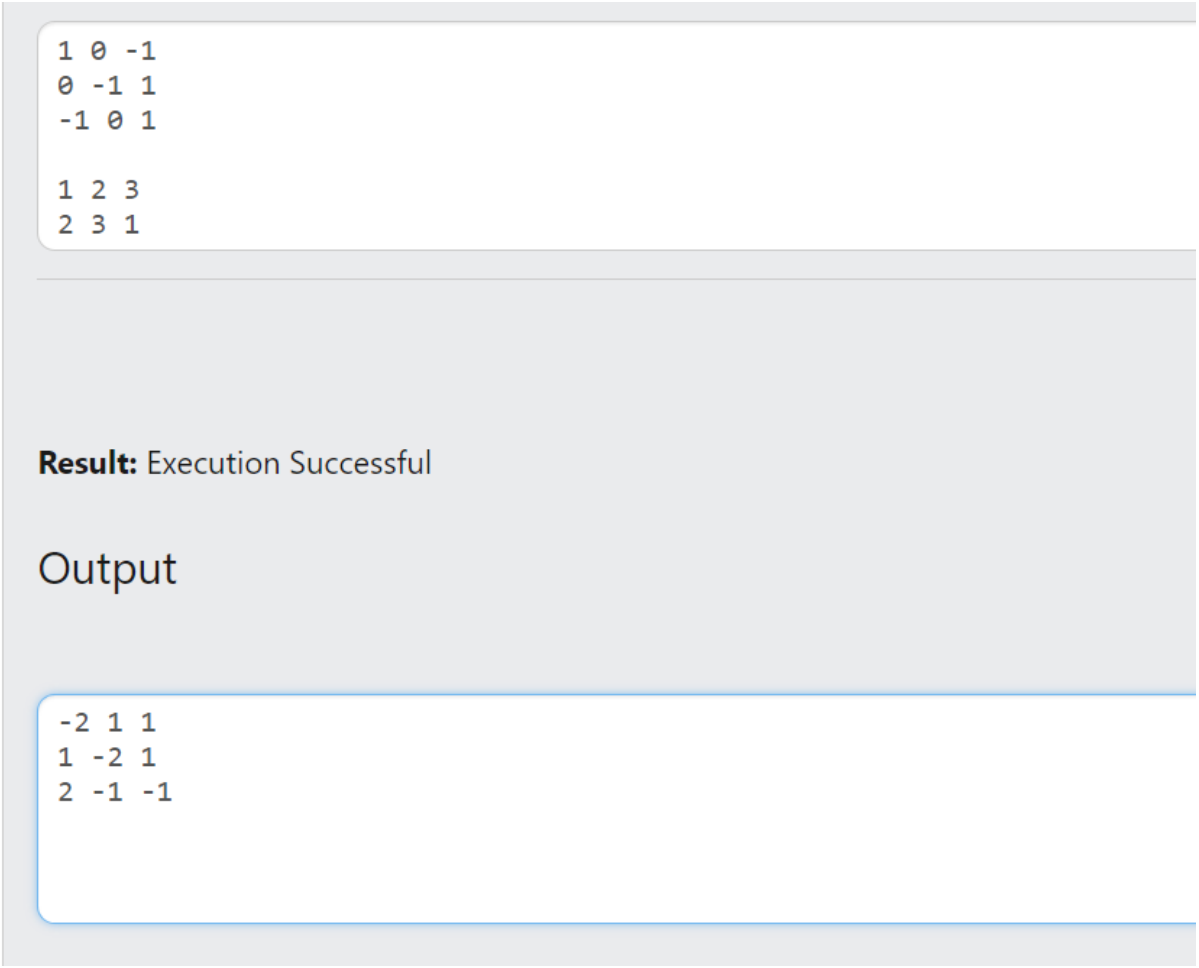
Aim: Write a Program to perform Matrix Multiplication.

Algorithm:
1) Create 3 2d array namely A,B,C in main loop.
2) Use 2 nested for loop to get values for matrix A and B.
3) Using 3 nested for loop we are performing matrix multiplication.
4) The value is then stored in C and output is printed.

Code:

```c
#include <stdio.h>

int main() {
    // Define matrices A and B

    int A[3][3];
    int B[3][3];
    int C[3][3];
    for(int i=0; i<3; i++) {
        for(int j=0; j<3; j++) {
            scanf("%d", &A[i][j]);
        }
    }

    for(int t=0; t<3; t++) {
        for(int y=0; y<3; y++) {
            scanf("%d", &B[t][y]);
        }
    }
    // Perform matrix multiplication
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            int sum = 0;
            for(int k = 0; k < 3; k++) {
                sum += A[i][k] * B[k][j];
            }
            C[i][j] = sum;
        }
    }

    // Print the resultant matrix
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            printf("%d ", C[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Output:


1 0 -1
0 -1 1
-1 0 1
1 2 3
2 3 1
Result: Execution Successful
Output
-2 1 1
1 -2 1
2 -1 -1

Ex 2. [Practice] Matrix Multiplication - Variable

Github Commit date: 09-07-2023

Aim: Write a Program to perform Matrix Multiplication.

Algorithm:
1) We are getting the size of the matrix from the user with variables "m" & "n".
2) Create 3 2d array namely A,B,C in main loop.
3) Use 2 nested for loop to get values for matrix A and B.
4) Using 3 nested for loop we are performing matrix multiplication.
5) The value is then stored in C and output is printed.

Code:

```c
#include <stdio.h>

int main() {
  int m, n, i, j, k;
  scanf("%d %d", &m, &n);

  int A[m][n], B[m][n], C[m][n];


  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      scanf("%d", &A[i][j]);
    }
  }


  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      scanf("%d", &B[i][j]);
    }
  }

  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      C[i][j] = 0;
      for (k = 0; k < n; k++) {
        C[i][j] += A[i][k] * B[k][j];
      }
    }
  }

  for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++) {
      printf("%d ", C[i][j]);
    }
    printf("\n");
  }

  return 0;
}
```

Output:

```
3
3
1 0 -1
0 -1 1
-1 0 1
```

Result: Execution Successful

Output

```
-2 1 1
1 -2 1
2 -1 -1
```

Ex 3. [Practice] Pattern - * Half Pyramid

Github Commit date: 09-07-2023

Aim: Write a c Program to print Half Pyramid pattern based on the input for number of rows from the user.

Algorithm:
1) Create a variable for the user to input value for number of rows and make constrains from 0 to 25 range of user input.
2) Use 2 for loop to print the pyramid with a newline keyword for every huydration.
3) Print "Invalid Input" if the user fails the constrains.

Code:

```c
#include <stdio.h>
int main(){
    int a,i,j;
    scanf("%d",&a);
    if(0<a && a<25){
        for(i=1;i<=a;i++){
            for(j=1;j<=i;j++){
                printf("*   ");
            }
            printf("\n");
        }
        return 0;
    }
    else{
        printf("Invalid Input");
    }
}
```

Output:

```
5
```

Result: Execution Successful

Output

```
*
*   *
*   *   *
*   *   *   *
*   *   *   *   *
```

Ex 4. [Practice] Pattern - * Full Pyramid

Github Commit date: 09-07-2023

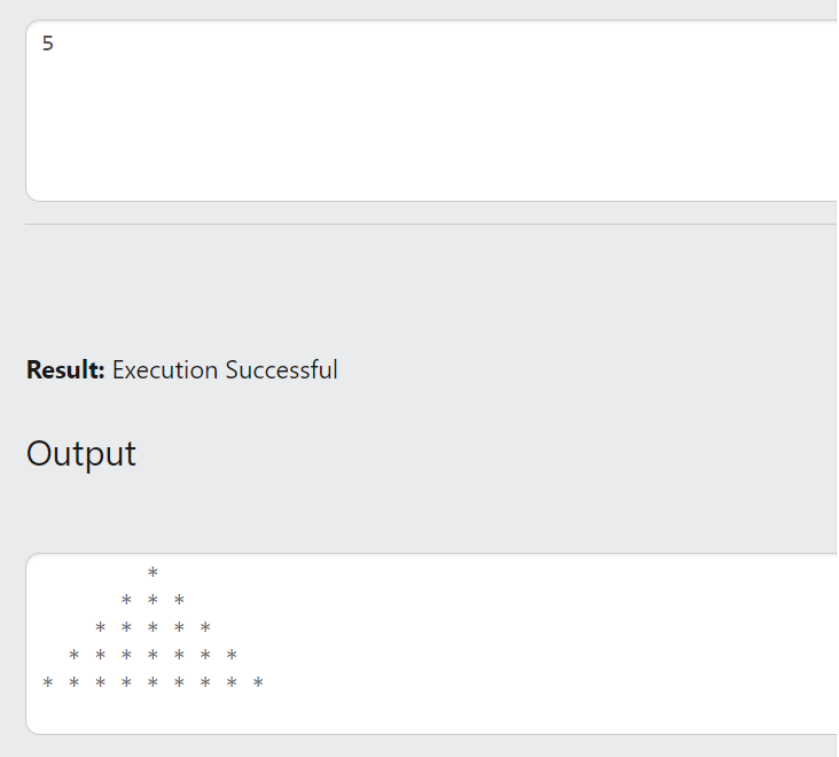
Aim: Write a c Program to print Full Pyramid based on the input for number of rows from the user.

Algorithm:
1) Create a variable for the user to input value for number of rows and make constrains from 0 to 25 range of user input.
2) Use 3 "for" loop, for the 1st loop is used to print the space for the pyramid and 2nd loop is to print the pyramid.
3) Print "Invalid Input" if the user fails the constrains.

Code:

```c
#include <stdio.h>
int main(){
    int a,i,j,k;
    scanf("%d",&a);
    if(0<a && a<25){
        for(i=1;i<=a;i++){
            for(k=1;k<=a-i;k++){
                printf("  ");
            }
            for(j=1;j<=2*i-1;j++){
                printf("* ");
            }
            printf("\n");
        }
        return 0;
    }
    else{
        printf("Invalid Input");
    }
}
```

Output:



Ex 5. [Quiz 1] Palindromic Pattern Half Pyramid

Github Commit date: 10-07-2023

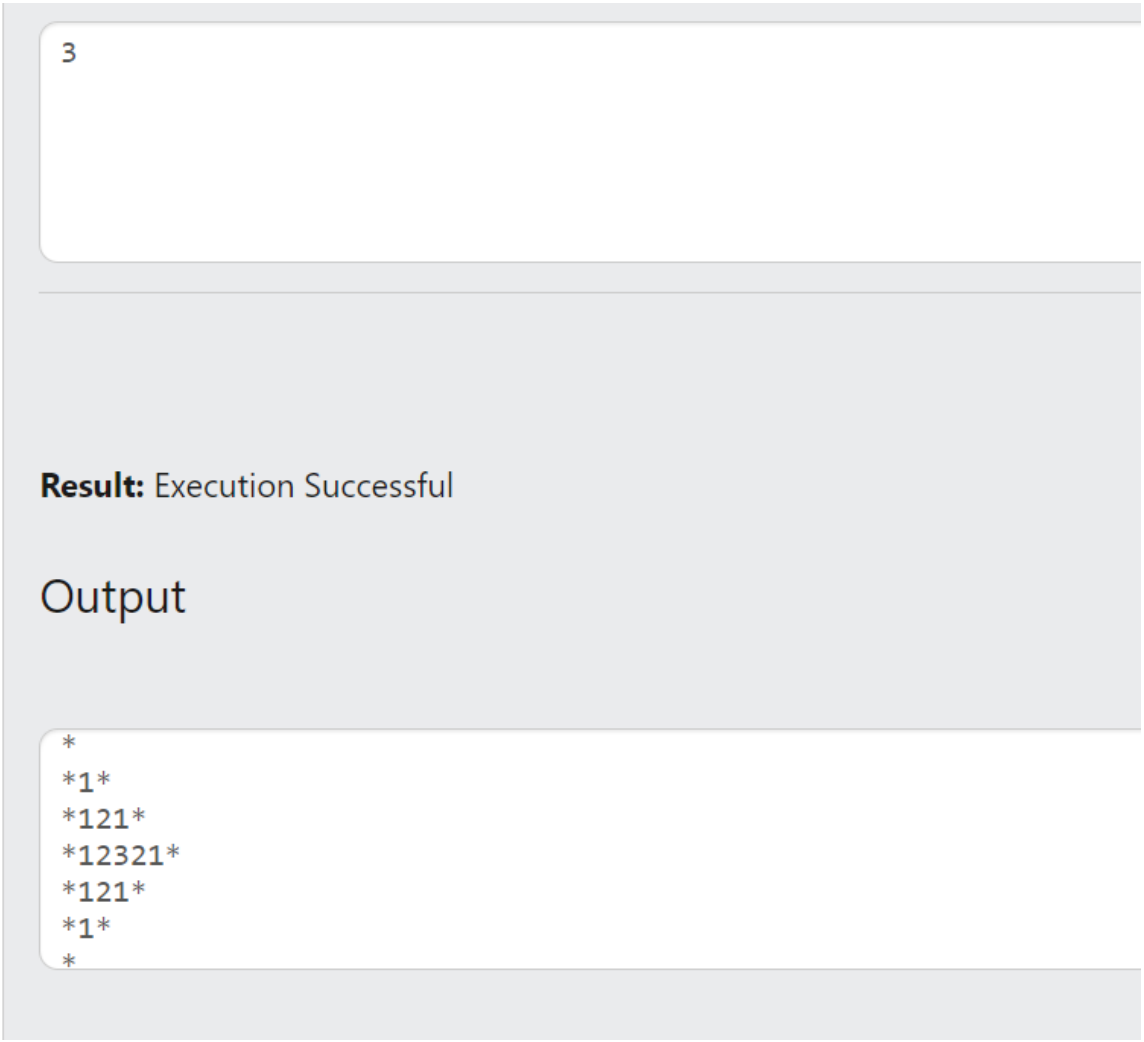
Aim: Write a C Program to print Palindromic Pattern Half Pyramid.

Algorithm:
1) Create a variable "a" for the user to input the number of rows.
2) 2 sets of nested for loop is used in this program.
3) The first set of loop is used to print the top right triangle, & second loop is used to print the bottom right triangle.
4) The numbers has been used to hytrate among them to form the following pattern.

Code:

```
#include <stdio.h>
int main() {
    int a,j;
    scanf("%d",&a);
    printf("*\n");
    for(int i=1;i<a+1;i++){
        printf("*");
        for(j=1;j<i+1;j++){
            printf("%d",j);
        }
        if(j!=2){
            j=j-1;
            do{
                j=j-1;
                printf("%d",j);
            }while(j!=1);
        }
        printf("*\n");
    }

    for(int i=a;i>1;i--){
        printf("*");
        for(j=1;j<i;j++){
            printf("%d",j);
        }
        if(j!=2){
            j=j-1;
            do{
                j=j-1;
                printf("%d",j);
            }while(j!=1);
        }
        printf("*\n");
    }
    printf("*");
}
```

Output:



Ex 6. [Quiz 1] Palindrome Check

Github Commit date: 09-07-2023

Aim: Write a C Program to check if the given number is Palindrome.

Algorithm:
1) Create a variable "a" for the user to enter the number to check if it is palindrome.
2) Using log to find the lenght of the number.
3) assigning it to another variable in an array.
4) Now using "for" loop check if the first element is equal to last element.
5) if equal it is palindrome else not palindrome.

Code:

```c
#include <stdio.h>
#include <math.h>
int main(){
    int a,k,ans=0,i,j;
    scanf("%d",&a);
    if (a<0){
        a=a*-1;
    }

    k=log10(a)+1;
    int val[k];
    for(i=0;i<k;i++){
        val[i]=a%10;
        a=a/10;
    }
    for(i=0;i<k/2;i++){
            for(j=k/2;j<k;j++){
                if (val[i]==val[j]){
                    ans=ans+1;
                    break;
                }
                else{
                    continue;
                }
            }
        }
        if((k/2)==ans){
            printf("Is a palindrome.");
        }
        else{
            printf("Is not a palindrome.");
        }
}
```

Output:

```
12321
```

Result: Execution Successful

Output

```
Is a palindrome.
```

```
1231
```

Result: Execution Successful

Output

```
Is not a palindrome.
```

Ex 7. [Practice] Addition and Subtraction of Complex Numbers using Structure

Github Commit date: 09-07-2023

Aim: Write a C Program to create a Structure that represents a complex number (a + ib), then receive the inputs for two complex numbers and perform addition & subtraction operation on them.

Algorithm:
1) Create an structure name complex_number with a array size of 4 with integer elements "real" and "img".
2) get user input for real and img part with index value 0 and 1.
3) now add the real part and img part, store it in the structure with index value 2.
4) subtraction real part and img part store it in the structure with index value 3.
5) now print the real and img part with the format "%d+%di".

Code:

```c
#include <stdio.h>

struct complex_number{
    int real;
    int img;
}cn[4];

int main(){
    scanf("%d",&cn[0].real);
    scanf("%d",&cn[0].img);

    scanf("%d",&cn[1].real);
    scanf("%d",&cn[1].img);

    //addition
    cn[2].real=cn[0].real+cn[1].real;
    cn[2].img=cn[0].img+cn[1].img;

    //subbraction
    cn[3].real=cn[0].real-cn[1].real;
    cn[3].img=cn[0].img-cn[1].img;

    printf("%d+%di\n",cn[2].real,cn[2].img);
    printf("%d+%di",cn[3].real,cn[3].img);
}
```

Output:

```
4 5
10 -2
```

Result: Execution Successful

Output

```
14+3i
-6+7i
```

Ex 8. 20CYS181 - Periodicals 1: Calculate Electricity Consumption

Github Commit date: 10-07-2023

Aim: Write a Program to Calculate Electricity Consumption.

Algorithm:
1) declare variables as array with size 5 customer_name, customer_eb_number, customer_units_consumed, and customer bill.
2) using if else statment calculate bill for the coustmer.
3) use for loop to print the output of the coustmers.

Code:

```
#include <stdio.h>
int main() {
  char customer_name[5][25];
  int customer_eb_number[5];
  float customer_units_consumed[5];
  float customer_bill[5];

  for (int i = 0; i < 5; i++) {

    printf("Enter the name of the customer: ");
    scanf("%s", customer_name[i]);
    printf("Enter the EB number of the customer: ");
    scanf("%d", &customer_eb_number[i]);
    printf("Enter the units consumed by the customer: ");
    scanf("%f", &customer_units_consumed[i]);

    customer_bill[i] = 0;
    if (customer_units_consumed[i] <= 100) {
      customer_bill[i] = customer_units_consumed[i] * 0;
    } else if (customer_units_consumed[i] <= 400) {
      customer_bill[i] = 100 * 0 + (customer_units_consumed[i] - 100) *
2.25;
    } else if (customer_units_consumed[i] <= 500) {
      customer_bill[i] = 100 * 0 + 300 * 2.25 +
(customer_units_consumed[i] - 400) * 4.50;
    } else if (customer_units_consumed[i] <= 600) {
      customer_bill[i] = 100 * 0 + 300 * 2.25 + 100 * 4.50 +
(customer_units_consumed[i] - 500) * 6;
    } else {
      customer_bill[i] = 100 * 0 + 300 * 2.25 + 100 * 4.50 + 100 * 6 +
(customer_units_consumed[i] - 600) * 8;
    }
  }

  for (int i = 0; i < 5; i++) {
    printf("Customer name: %s\n", customer_name[i]);
    printf("EB number: %d\n", customer_eb_number[i]);
    printf("Units consumed: %.3f\n", customer_units_consumed[i]);
    printf("Bill: INR %.2f\n", customer_bill[i]);
  }
  return 0;
}
```

Output:

```
Enter the name of the customer: amal
Enter the EB number of the customer: 1
Enter the units consumed by the customer: 12
Enter the name of the customer: ananth
Enter the EB number of the customer: 2
Enter the units consumed by the customer: 14
Enter the name of the customer: shravan
Enter the EB number of the customer: 3
Enter the units consumed by the customer: 11
Enter the name of the customer: krishna
Enter the EB number of the customer: 4
Enter the units consumed by the customer: 13
Enter the name of the customer: hema
Enter the EB number of the customer: 5
Enter the units consumed by the customer: 15
Customer name: amal
EB number: 1
Units consumed: 12.000
Bill: INR 123.23
Customer name: ananth
EB number: 2
Units consumed: 14.000
Bill: INR 130.43
```

```
Customer name: shravan
EB number: 3
Units consumed: 11.000
Bill: INR 112.89
Customer name: krishna
EB number: 4
Units consumed: 13.000
Bill: INR 127.34
Customer name: hema
EB number: 5
Units consumed: 15.000
Bill: INR 136.39
```

Ex 9. 20CYS181 - Periodicals 1: Number Diamond Pattern

Github Commit date: 10-07-2023

Aim: Write a C Program to print the below numbered diamond pattern.

Algorithm:

1) Create a variable "a" for the user to input the value of rows.
2) we will be using 2 sets of nested "for" loop.
3) the first nested "for" loop will be used to create an full pyramid with numbers.
4) in second loop an inverted full pyramid is printed.

Code:

```c
#include <stdio.h>
int main(){
    int a,i,j,k,con=1,val;
    scanf("%d",&a);
    for(i=0;i<a;i++){
        for(j=0;j<a*2-i*2;j++){
            printf(" ");
        }
        val=9;
        for(k=0;k<con;k++){
            printf("%d ",val);
            val--;
        }
        printf("\n");
        con=con+2;
    }

    con=con-4;

    for(i=a-2;i>=0;i--){
        for(j=0;j<a*2-i*2;j++){
            printf(" ");
        }
        val=9;
        for(k=0;k<con;k++){
            printf("%d ",val);
            val--;
        }
        printf("\n");
        con=con-2;
    }
}
```

Output:

```
3
```

Result: Execution Successful

Output

```
    9
  9 8 7
9 8 7 6 5
  9 8 7
    9
```

Ex 10. 20CYS181 - Periodicals 1: Alphabet Hourglass Pattern

Github Commit date: 10-07-2023

Aim: Write a C Program to print the below alphabet hourglass pattern.

Algorithm:

1) Create a variable "a" for the user to input the value of rows.
2) we will be using 2 sets of nested "for" loop.
3) the first nested "for" loop will be used to create an inverted full pyramid is printed.
4) in second loop an full pyramid is printed.
5) we will be using ascii value to access alphabets "D" ascii value is 68 and it will be incremented by 1 for each hydration .

Code:

```c
#include <stdio.h>
int main() {
    int i,j,k,a,con;
    scanf("%d",&a);
    con=a*2-1;
    for(i=a;i>0;i--){
        printf(" ");
        for(j=0;j<a*2-i*2;j++){
            printf(" ");
        }
        int val=68;
        for(k=0;k<con;k++){
            printf("%c ",val);
            val++;
        }
        con=con-2;
        printf("\n");
    }

    con=con+4;
    for(i=0;i<a-1;i++){
        for(j=0;j<(a*2-i*2)-3;j++){
            printf(" ");
        }
        int val=68;
        for(k=0;k<con;k++){
            printf("%c ",val);
            val++;
        }
        printf("\n");
        con=con+2;
    }
}
```

Output:

```
4
```

Result: Execution Successful

Output

```
D E F G H I J
  D E F G H
    D E F
      D
    D E F
  D E F G H
D E F G H I J
```

Ex 11. [Periodicals 2] Simple Student Management System

Github Commit date: 09-07-2023

Aim: You are asked to create a program that simulates a Simple Student Management System. The system should allow the user to perform various operations on a list of students using a menu-driven interface. The program should utilize structs, enums, and menu-driven programming.

Algorithm:
1) create structures with name Student Contains the following attributes:
name (string), age (integer),score (enum).
2) Implement a menu-driven program that offers the following options: Add a student, Display all students, Find the highest-scoring student, Exit the program.
3) these functions are used to print output: addStudent, displayStudents, findHighestScoringStudent.

Code:

[Simple Student Management System]

Output:

```
1
Ramaguru
33
A
4
```

Result: Execution Successful

Output

```
Student added successfully.
Exiting the program. Thank you for using our system!
```

Ex 12. [Practice][Functions] Generate all the prime numbers between two given numbers

Github Commit date: 09-07-2023

Aim: Write a C Program to print all the prime numbers between two given numbers. generatePrimes function should be used to generate prime numbers and isPrime function should check the prime status for each number.

Algorithm:
1) Create varibles "a" and "b" for the user to enter the range of numbers to be checkeded for prime numbers.
2) create a function generate_prime with attributes a and b.
3) make a "for" loop to generate all numbers between "a" and "b" save it in a array "val".
4) print the prime numbers in array "val" by using "isprime" function which returns 1 if prime or return 0 if false.
5) The prime numbers will be printed between "a" and "b"

Code:

```c
#include <stdio.h>
int main(){
    int a,b;
    scanf("%d %d",&a,&b);
    generate_prime(a,b);
}

void generate_prime(int a,int b){
    int k = b-a,i,check;
    int it = a;
    int val[k];
    for(i=0;i<k;i++){
        val[i]=a;
        a=a+1;
    }
    printf("Prime numbers between %d and %d are: ",it,b);
    for(i=0;i<k;i++){
        check=isprime(val[i]);
        if (check==1){
            printf("%d ",val[i]);
        }
    }
}

int isprime(int a){
    int ans=1,i;
    for(i=1;i<a;i++){
        if (a%i==0){
            ans=ans+1;
        }
    }
    if (ans==2){
        return 1;
    }
    else{
        return 0;
    }
}
```

Output:

```
5 100
```

Result: Execution Successful

Output

```
Prime numbers between 5 and 100 are: 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

Ex 13. [Practice][Functions] Convert Decimal to Binary, Octal and Hexadecimal

Github Commit date: 09-07-2023

Aim: Implement 3 functions decimalToBinary, decimalToOctal, decimalToHexadecimal to convert a given decimal number (>0) to Binary, Octal, and Hexadecimal.

Algorithm:
1) Create 3 functions namely "decimaltobinary", "decimaltooctal", "decimaltohexadecimal".
2) declare a variable "a" for the user to input.
3) add constrains that "a" should be greater than 0.
4) calling all 3 functions for "a".
5) octal is converted by using "%o" and for hexadecimal is converted by using "%x".
6) the output is then printed.

Code:

```c
#include <stdio.h>
#include <math.h>
int main(){
    int a;
    scanf("%d",&a);
    if(a<=0){
        printf("Error: Value should be greater than 0");
    }
    else{
        decimalToBinary(a);
        printf("\n");
        decimalToOctal(a);
        printf("\n");
        decimalToHexadecimal(a);
    }
}

void decimalToBinary(int a){
    int val[30],k=0,i;
    while(a!=0){
        if(a%2==0){
            val[k]=0;
            k=k+1;
        }
        else{
            val[k]=1;
            k=k+1;
        }
        a=a/2;
    }
    printf("Binary equivalent: ");
    for(i=k-1;i>=0;i--){
        printf("%d",val[i]);
    }
}

void decimalToOctal(int a){
    int val[30],k=0,i,it;
    while(a!=0){
        it=(a%8);
        val[k]=it;
        k=k+1;
```

```
        a=a/8;
    }
    printf("Octal equivalent: ");
    for(i=k-1;i>=0;i--){
        printf("%d",val[i]);
    }
}

void decimalToHexadecimal(int a){
    printf("Hexadecimal equivalent: ");
    printf("%X",a);
}
```

Output:

```
2023
```

Result: Execution Successful

Output

```
Binary equivalent: 11111100111
Octal equivalent: 3747
Hexadecimal equivalent: 7E7
```

Ex 14. [Practice] Calculate Area - Structs, Unions, and Enums

Github Commit date: 09-07-2023

Aim: Write a C program that calculates area of circle and rectangle using structs, union, enums

Algorithm:

1) Create a structure shape with union dimension and enum type.
2) declare a variable "a" for menu driven value.
3) for case 1 is to get value form user to calculate the area of circle.
4) the values of the radious is stored in struct "s.d.radius".
5) for case 2 is to get value from user to calculate the are of rectangle.
6) the values are stored in struct "s.d.length[0]" and "s.d.length[1]".
7) the output is printed in the formate "Area of circle: %.4f units" and "Area of rectangle: %.4f units".

Code:

```c
#include <stdio.h>

struct shape{
    union dimensions{
        float length[2];
        float radius;
    }d;
    enum type{
        circle,rectangle
    }e;
}s;

int main(){
    int a;
    float ans;
    scanf("%d",&a);
    if(0<a && a<3){
        switch (a){
            case 1:
                scanf("%f",&s.d.radius);
                ans=3.14*s.d.radius*s.d.radius;
                printf("Area of the circle: %.4f units",ans);
                break;
            case 2:
                scanf("%f",&s.d.length[0]);
                scanf("%f",&s.d.length[1]);
                ans=s.d.length[0]*s.d.length[1];
                printf("Area of the rectangle: %.4f units",ans);
                break;
        }
    }
    else{
        printf("Invalid choice!");
    }
}
```

Output:

```
1 24
```

Result: Execution Successful

Output

```
Area of the circle: 1808.6400 units
```

Ex 15. [Practice] Employee Management System

Github Commit date: 10-07-2023

Aim: Write a C program to implement an Employee Management System.

Algorithm:
1) The program starts by declaring the necessary data structures and variables.
2) The user is prompted to enter the number of employees (numEmployees).
3) A loop is initiated to input employee details for each employee.Within the loop, the program asks
4) the user to enter the employee's name and age.
5) The program then asks for the employee type (employeeType) using an integer input.
6) 0 represents FullTime, and 1 represents PartTime.After inputting the employee details for all employees, the program prompts the user to choose an operation from the menu.
7) then the output is printed.

Code:

[Employee Management System]

Output:

```
/tmp/kmdAoOdADW.o
Enter the number of employees: 2
Enter details for Employee 1
Name: amal
Age: 18
Employee Type (0 for Full-time, 1 for Part-time): 0
Monthly Salary: 12000
Bonus: 120
Enter details for Employee 2
Name: shravan
Age: 19
Employee Type (0 for Full-time, 1 for Part-time): 1
Hourly Rate: 1200
Hours Worked: 36
```

```
Employee Management System
1. Calculate total monthly salary for all full-time employees
2. Calculate total earnings for all part-time employees
3. Display details of all employees
4. Exit
Enter your choice: 3
Details of all employees:
Employee 1
Name: amal
Age: 18
Employee Type: Full-time
Monthly Salary: 12000.00
Bonus: 120.00
--------------------
Employee 2
Name: shravan
Age: 19
Employee Type: Part-time
Hourly Rate: 1200.00
Hours Worked: 36
--------------------
```