----------Works----------

1(grade validator)

```
#include<stdio.h>
int no_of_marks=5;
volatile int external_mark=2;
int input_grade(){
   int arr[no_of_marks];
   float total=0;
   for(int i=1;i<no_of_marks+1;i++){
      printf("enter the mark %d :",i);
      scanf("%d",&arr[i]);
      if(external_mark!=0){
         arr[i]+=external_mark;
      }
      total+=arr[i];
   }
   float avg_marks=(total/no_of_marks);
   printf("applying external mark update of %d for each marks:\n",external_mark);
   printf("the average marks= %f\n",avg_marks);
   return avg_marks;
}
void determine_grade(float grade) {
   if (grade >= 90.0) {
      printf("Student has an overall grade of A.\n\n");
   } else if (grade >= 80.0) {
      printf("Student has an overall grade of B.\n\n");
   } else if (grade >= 70.0) {
```

```
        printf("Student has an overall grade of C.\n\n");
    } else if (grade >= 60.0) {
        printf("Student has an overall grade of D.\n\n");
    } else if (grade >= 50.0) {
        printf("Student has an overall grade of E.\n\n");
    } else {
        printf("Fail\n");
    }
}
int main(){
    static int total_students_processed=1;
    printf("enter the number of students to process :\n");
    scanf("%d",&total_students_processed);
    for(int i=1;i<total_students_processed+1;i++){
        printf("enter the datas for student %d\n",i);
        float result=input_grade();
        determine_grade(result);
    }
    return 0;
}
```

2(Prime number checker)

```
#include<stdio.h>
#include<stdbool.h>
int prime(int n){
    if(n%2==0){
        return false;
    }else{
```

```
        return true;
    }
}

int main(){
    int n;
    printf("enter a number \n");
    scanf("%d",&n);
    printf("the prime numbers between 1 and %d are :\n",n);
    for(int i=1;i<=n;i++){
        if(prime(i)){
            printf("%d\n",i);
        }
    }
}
```

3(calculator)

```
#include<stdio.h>
static int no_of_operations=0;
int main(){
   int a,b,user_input;
   do{
      printf("Enter the 2 numbers :\n");
      scanf("%d%d",&a,&b);
      printf("enter the operation to perform\n1.add\n2.subtract\n3.multiply\n4.divide\n5.Exit:");
      scanf("%d",&user_input);
      switch(user_input){
          case 1:
```

```
            printf("the sum is %d\n\n",a+b);
            no_of_operations+=1;
            break;
        case 2:
            printf("the differnece is %d\n\n",a-b);
            no_of_operations+=1;
            break;
        case 3:
            printf("the product is %d\n\n",a*b);
            no_of_operations+=1;
            break;
        case 4:
            printf("the quotient is %d\n\n",a/b);
            no_of_operations+=1;
            break;
        default:
            printf("Exiting\n\n");
    }
   printf("the total number of operations performed is %d\n",no_of_operations);
  }while(user_input!=5);
}
```

4(matrix calculation)

```
#include <stdio.h>
#define MAX_SIZE 5
static int result_add[MAX_SIZE][MAX_SIZE];
static int result_mul[MAX_SIZE][MAX_SIZE];
```

```
void matrix_add(int mat1[MAX_SIZE][MAX_SIZE], int mat2[MAX_SIZE][MAX_SIZE], int rows, int cols) {
    if (rows > MAX_SIZE || cols > MAX_SIZE) {
        printf("Error: Matrix size exceeds maximum allowed size\n");
        return;
    }
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            result_add[i][j] = mat1[i][j] + mat2[i][j];
        }
    }
    printf("\nMatrix Addition Result:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", result_add[i][j]);
        }
        printf("\n");
    }
}
void matrix_multiply(int mat1[MAX_SIZE][MAX_SIZE], int mat2[MAX_SIZE][MAX_SIZE], int rows1, int
cols1, int rows2, int cols2) {
    if (cols1 != rows2) {
        printf("Error: Matrix dimensions are incompatible for multiplication\n");
        return;
    }
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            result_mul[i][j] = 0;
        }
    }
```

```
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            for (int k = 0; k < cols1; k++) {
                result_mul[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
    printf("\nMatrix Multiplication Result:\n");
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            printf("%d ", result_mul[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int mat1[MAX_SIZE][MAX_SIZE], mat2[MAX_SIZE][MAX_SIZE];
    int rows1, cols1, rows2, cols2;
    printf("Enter the number of rows and columns for the first matrix: ");
    scanf("%d %d", &rows1, &cols1);
    printf("Enter the elements of the first matrix:\n");
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols1; j++) {
            scanf("%d", &mat1[i][j]);
        }
    }
    printf("Enter the number of rows and columns for the second matrix: ");
    scanf("%d %d", &rows2, &cols2);
```

```
    printf("Enter the elements of the second matrix:\n");
    for (int i = 0; i < rows2; i++) {
        for (int j = 0; j < cols2; j++) {
            scanf("%d", &mat2[i][j]);
        }
    }
    if (rows1 == rows2 && cols1 == cols2) {
        matrix_add(mat1, mat2, rows1, cols1);
    } else {
        printf("Error: Matrix dimensions are incompatible for addition\n");
    }
    matrix_multiply(mat1, mat2, rows1, cols1, rows2, cols2);

    return 0;
}
```

5(Temperature validator)

```
#include<stdio.h>
static int max=0;
int main(){
    int i=39;
    while(i>-10 && i<40){
        if(i>max){
            max=i;
        }
        printf("The current temperature is %d\n",i);
        if(i>30){
            printf("the temperature is too hot!\n");
```

```
        }else if(i<30 && i>20){
            printf("the temperature is ideal for outdoors!\n");
        }else if(i<20 && i>10){
            printf("the temperature is chill!\n");
        }else{
            printf("the temperature is too cold!\n");
        }
    i-=1;
    printf("the max temperature recorded is %d",max);
}}
```

6(Password validator)

```
#include<stdio.h>
#include<ctype.h>
#include<stdbool.h>
#include<string.h>

bool validate_password(char password[]) {
    int alpha = 0, digit = 0;
    int length = strlen(password);

    if (length >= 4 && length <= 8) {
        for (int i = 0; i < length; i++) {
            if (isalpha(password[i])) {
                alpha += 1;
            } else if (isdigit(password[i])) {
                digit += 1;
            }
```

```
        }

        if (alpha > 0 && digit > 0) {
            return true;
        } else {
            return false;
        }
    } else {
        return false;
    }
}

int main() {
    char password[100];
    int user_input;
    do {
        printf("Enter the password: ");
        scanf("%s", password);
        bool result = validate_password(password);
        if (result) {
            printf("The password is valid!\n");
        } else {
            printf("Not a valid password.\n");
        }
        printf("Enter '1' to continue or any other key to stop: ");
        scanf("%d", &user_input);
    } while (user_input == 1);

    return 0;
```

```
}
```

7(Bank functionality)

```
#include <stdio.h>

int main() {
    static double balance = 1000.00;
    const double MAX_WITHDRAWAL_LIMIT = 500.00;
    int transactionType;
    double amount;
    char continueTransaction;
    printf("Welcome to the Bank!\n");
    printf("Your initial balance is: Rs.%.2f\n", balance);
    do {
        printf("\nSelect transaction type:\n");
        printf("1. Deposit\n");
        printf("2. Withdraw\n");
        printf("Enter your choice (1 or 2): ");
        scanf("%d", &transactionType);
        if (transactionType == 1) {
            printf("Enter amount to deposit: Rs.");
            scanf("%lf", &amount);
            if (amount > 0) {
                balance += amount;
                printf("You have successfully deposited $%.2f\n", amount);
                printf("New balance: Rs.%.2f\n", balance);
            } else {
```

```
            printf("Invalid deposit amount.\n");
        }
    }
    else if (transactionType == 2) {
        printf("Enter amount to withdraw: Rs.");
        scanf("%lf", &amount);
        if (amount > 0 && amount <= balance) {
            if (amount <= MAX_WITHDRAWAL_LIMIT) {
                balance -= amount;
                printf("You have successfully withdrawn Rs.%.2f\n", amount);
                printf("New balance: Rs.%.2f\n", balance);
            } else {
                printf("Error: The withdrawal amount exceeds the maximum limit of Rs.%.2f\n",
MAX_WITHDRAWAL_LIMIT);
            }
        } else if (amount > balance) {
            printf("Error: Insufficient funds.\n");
        } else {
            printf("Invalid withdrawal amount.\n");
        }
    } else {
        printf("Invalid transaction type.\n");
    }
    printf("\nDo you want to perform another transaction? (y/n): ");
    getchar();
    scanf("%c", &continueTransaction);

  } while (continueTransaction == 'y' || continueTransaction == 'Y');
```

```
    printf("\nThank you for using our service. Your final balance is Rs.%.2f\n", balance);

    return 0;
}
```

8(Clock simulation)

```
#include<stdio.h>
#include<unistd.h>

volatile int tick=0;
static int total_ticks=0;

void run_clock(){
    volatile char ch='A';
    int hours=0,minutes=0,seconds=0;
    while(1){
        if(++tick){
            total_ticks+=1;
            seconds+=1;
            if(seconds>=60){
                minutes+=1;
                seconds=0;
                if(minutes>=60){
                    hours+=1;
                    minutes=0;
                    if(hours>12 && hours<24){
                        hours=0;
                        ch='P';
```

```
            }else if(hours>24){
                hours=0;
                ch='A';
            }
          }
        }
        printf("\r%02d:%02d:%02d:%c", hours, minutes, seconds,ch);
        fflush(stdout);
        sleep(1);

      }

   }

}

int main(){
   printf("digital clock\n");
   run_clock();
   return 0;
}
```

9.(Game score tracker)

```
#include <stdio.h>

int main() {
   static int current_score = 0;
```

```
    const int winning_score = 10;
    int score_change;
    char play_again;
    printf("Welcome to the Game Score Tracker!\n");
    while (current_score < winning_score) {
        printf("\nCurrent Score: %d\n", current_score);
        printf("Enter score change for this round (positive for winning points, negative for losing points): ");
        scanf("%d", &score_change);
        current_score += score_change;
        if (current_score < 0) {
            printf("Score cannot be negative! Setting score to 0.\n");
            current_score = 0;
        }
        if (current_score >= winning_score) {
            printf("\nCongratulations! You've won with a score of %d!\n", current_score);
            break;
        } else {
            printf("Current score: %d\n", current_score);
            printf("Do you want to continue playing? (y/n): ");
            scanf(" %c", &play_again);
            if (play_again == 'n' || play_again == 'N') {
                printf("\nGame Over. Final Score: %d\n", current_score);
                break;
            }
        }
    }

    return 0;
}
```