Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

Question 1:

Create a program that validates password strength based on these rules:

- At least 8 characters long
- Contains at least one uppercase letter
- Contains at least one digit
- Contains at least one special character (!, @, #, $, %)

```c
#include <stdio.h>
#include <string.h>

int validatePassword(char password[100]);

int main() {
    char password[100];
    printf("Enter a password to validate: ");
    fgets(password, sizeof(password), stdin);
    password[strcspn(password, "\n")] = 0;
    if (validatePassword(password)) {
        printf("Password is strong and meets all requirements.\n");
    } else {
        printf("Password is weak.\n");
    }

    return 0;
}

int validatePassword(char password[100])
{
    int length = strlen(password), hasUpperCase = 0, hasDigit = 0, hasSpecialChar = 0, i;
    if (length < 8)
    {
        printf("Password must be at least 8 characters long.\n");
        return 0;
    }
    for (i = 0; i < length; i++)
    {
        if(isupper(password[i]))
        {
            hasUpperCase = 1;
        }
        else if(isdigit(password[i]))
        {
            hasDigit = 1;
        }
        else if(password[i] == '!' || password[i] == '@' || password[i] == '#' || password[i] == '$' || password[i] == '%')
        {
```

Student Name: _____Wajiha Nazir_____ Roll No: _____25K-3021_____ Section: BSE-1B

```
            hasSpecialChar = 1;
        }
    }
    if (!hasUpperCase)
    {
        printf("Password must contain at least one uppercase letter.\n");
        return 0;
    }
    if (!hasDigit)
    {
        printf("Password must contain at least one digit.\n");
        return 0;
    }
    if (!hasSpecialChar)
    {
        printf("Password must contain at least one special character (!, @, #, $, %%).\n");
        return 0;
    }
    return 1;
}
```

```
Enter a password to validate: Abcf3J@mK
Password is strong and meets all requirements.

--------------------------------
Process exited after 27.41 seconds with return value 0
Press any key to continue . . .
```

Question 2:

Write a program that extracts and displays the domain from email addresses.

```
Email Domain Extraction:
Email: user@gmail.com              → Domain: gmail.com
Email: john.doe@company.org        → Domain: company.org
Email: invalid-email               → Domain: Invalid email
Email: admin@university.edu        → Domain: university.edu
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```c
#include <stdio.h>
#include <string.h>

int main()
{
    char email[50], choice;
    int length, i, found;
    printf("Email Domain Extraction:\n");
    do
    {
        printf("Email: ");
        fgets(email, sizeof(email), stdin);
        email[strcspn(email, "\n")] = '\0';
        found = 0;
        for(i = 0; email[i] != '\0'; i++)
        {
            if(email[i] == '@')
            {
                found = 1;
                printf("Domain: %s\n", email + i + 1);
                break;
            }
        }
        if(found == 0)
            printf("Invalid Email (No @ found)\n");
        printf("Do you Want to Continue (Y/N): ");
        scanf(" %c", &choice);
        while(getchar() != '\n');
    }
    while(choice == 'Y' || choice == 'y');
    return 0;
}
```

```
Email Domain Extraction:
Email: sahitowajiha@gmail.com
Domain: gmail.com
Do you Want to Continue (Y/N): y
Email: nazir.nazy@hotmail.com
Domain: hotmail.com
Do you Want to Continue (Y/N): Y
Email: tayyaba313
Invalid Email (No @ found)
Do you Want to Continue (Y/N): n

--------------------------------
Process exited after 27.14 seconds with return value 0
Press any key to continue . . .
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

Question 3:

Write a program that counts vowels, consonants, digits, and spaces in a string.

```
Text: "Hello World 123! Programming is fun."

Statistics:
Vowels: 9
Consonants: 18
Digits: 3
Spaces: 6
Total characters: 37
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

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

int main()
{
    char text[100];
    int length, vowels=0, digits=0, space=0, consonent=0, i;

    printf("text: ");
    fgets(text, sizeof(text), stdin);
    text[strcspn(text, "\n")] = '\0';

    length = strlen(text);

    for(i = 0; text[i] != '\0'; i++)
    {
        if(text[i]=='a' || text[i]=='e' || text[i]=='i' || text[i]=='o' || text[i]=='u' ||
           text[i]=='A' || text[i]=='E' || text[i]=='I' || text[i]=='O' || text[i]=='U')
        {
            vowels++;
        }
        else if(isdigit(text[i]))
        {
            digits++;
        }
        else if(text[i] == ' ')
        {
            space++;
        }
        else if(isalpha(text[i]))
        {
            consonent++;
        }
    }

    printf("Statistics:\n");
    printf("Vowels: %d\n", vowels);
    printf("Consonents: %d\n", consonent);
    printf("Digits: %d\n", digits);
    printf("Spaces: %d\n", space);
    printf("Total Characters: %d\n", length);

    return 0;
}
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
text: "Hello World 123! Programming is fun."
Statistics:
Vowels: 8
Consonents: 18
Digits: 3
Spaces: 5
Total Characters: 38

--------------------------------
Process exited after 22.38 seconds with return value 0
Press any key to continue . . .
```

Question 4:

Create a program that encrypts a message by shifting each letter by 3 positions.

```
Original: Meet me at the park tomorrow
Encrypted: Phhw ph dw wkh sdun wrpruurz
Decrypted: Meet me at the park tomorrow
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

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

void encrypt(char text[], char encrypted[]);

void decrypt(char text[], char decrypted[]);

int main()
{
    char text[100], encrypted[100], decrypted[100];

    printf("Enter message: ");
    fgets(text, sizeof(text), stdin);
    text[strcspn(text, "\n")] = '\0';

    encrypt(text, encrypted);
    decrypt(encrypted, decrypted);

    printf("\nOriginal: %s\n", text);
    printf("Encrypted: %s\n", encrypted);
    printf("Decrypted: %s\n", decrypted);

    return 0;
}

void encrypt(char text[], char encrypted[])
{
    int i;
    for(i = 0; text[i] != '\0'; i++)
    {
        char ch = text[i];

        if(ch >= 'a' && ch <= 'z')
            encrypted[i] = ((ch - 'a' + 3) % 26) + 'a';

        else if(ch >= 'A' && ch <= 'Z')
            encrypted[i] = ((ch - 'A' + 3) % 26) + 'A';
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
        else
            encrypted[i] = ch;    // spaces, punctuation
    }
    encrypted[i] = '\0';
}

void decrypt(char text[], char decrypted[])
{
    int i;
    for(i = 0; text[i] != '\0'; i++)
    {
        char ch = text[i];

        if(ch >= 'a' && ch <= 'z')
            decrypted[i] = ((ch - 'a' - 3 + 26) % 26) + 'a';

        else if(ch >= 'A' && ch <= 'Z')
            decrypted[i] = ((ch - 'A' - 3 + 26) % 26) + 'A';

        else
            decrypted[i] = ch;
    }
    decrypted[i] = '\0';
}
```

```
Enter message: Meet me at the park tomorrow

Original: Meet me at the park tomorrow
Encrypted: Phhw ph dw wkh sdun wrpruurz
Decrypted: Meet me at the park tomorrow

--------------------------------
Process exited after 25.89 seconds with return value 0
Press any key to continue . . .
```

Question 5:

Create a program that stores student names and marks, then finds the highest scorer.

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
Student Marks:
Name            Marks
----------------------
Alice           85
Bob             92
Charlie         78
Diana           96
Eve             88

Highest Scorer: Diana with 96 marks
Average Marks: 87.80
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```c
#include <stdio.h>
#include <string.h>

int main()
{
    char names[5][20];
    int marks[5], i, highest = -1, highestIndex = 0, sum = 0;

    printf("Enter names and marks for 5 students:\n");

    for (i = 0; i < 5; i++)
    {
        printf("\nStudent %d Name: ", i + 1);
        scanf("%s", names[i]);

        printf("Marks: ");
        scanf("%d", &marks[i]);

        sum += marks[i];

        if (marks[i] > highest)
        {
            highest = marks[i];
            highestIndex = i;
        }
    }

    printf("\nStudent Marks:\n");
    printf("Name\t\tMarks\n");
    printf("-----------------------\n");

    for (i = 0; i < 5; i++)
    {
        printf("%-10s\t%d\n", names[i], marks[i]);
    }

    float average = sum / 5.0;

    printf("\nHighest Scorer: %s with %d marks\n", names[highestIndex], highest);
    printf("Average Marks: %.2f\n", average);

    return 0;
}
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
Student 1 Name: Alice
Marks: 85

Student 2 Name: Bob
Marks: 92

Student 3 Name: Charlie
Marks: 78

Student 4 Name: Diana
Marks: 96

Student 5 Name: Eve
Marks: 88

Student Marks:
Name            Marks
-----------------------
Alice           85
Bob             92
Charlie         78
Diana           96
Eve             88

Highest Scorer: Diana with 96 marks
Average Marks: 87.80

--------------------------------
Process exited after 37.93 seconds with return value 0
Press any key to continue . . .
```

Question 6:

You are a teacher who needs to manage student marks for a class. You have 5 students in your class, each with their name and test marks. You want to create a program that:

1. **Shows a list** of all students with their marks in a neat table
2. **Finds the top student** - who got the highest marks in the test
3. **Calculates the class average** - the average marks of all students

How the program works:

- It stores student names like "Alice", "Bob", "Charlie" etc.
- It stores their corresponding marks like 85, 92, 78 etc.
- It displays everything in a clean table format
- It goes through all marks to find which student has the highest score
- It adds up all marks and divides by number of students to find average

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

You write down all student names and their scores, then you figure out who got the highest marks and what the class average was.

```c
#include <stdio.h>

int main()
{
    char names[5][20] = {"Alice", "Bob", "Charlie", "Diana", "Eve"};
    int marks[5] = {85, 92, 78, 96, 88};
    int highestMarks = marks[0], i, highestIndex = 0, sum = 0;
    printf("Student Marks:\n");
    printf("Name\t\tMarks\n");
    printf("-------------------------\n");
    for (i = 0; i < 5; i++)
    {
        printf("%-10s\t%d\n", names[i], marks[i]);
        sum += marks[i];

        if (marks[i] > highestMarks)
        {
            highestMarks = marks[i];
            highestIndex = i;
        }
    }
    float average = sum / 5.0;
    printf("\nHighest Scorer: %s with %d marks\n", names[highestIndex], highestMarks);
    printf("Class Average: %.2f\n", average);
    return 0;
}
```

```
Student Marks:
Name            Marks
-------------------------
Alice           85
Bob             92
Charlie         78
Diana           96
Eve             88

Highest Scorer: Diana with 96 marks
Class Average: 87.80

--------------------------------
Process exited after 0.06159 seconds with return value 0
Press any key to continue . . .
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

Question 7:

You are a school administrator managing student course registrations. You have 4 students who have signed up for different courses. The program should:

1. **Show each student's registered courses** - display what courses each student is taking
2. **Find students taking a specific course** - like finding all students taking Physics
3. **Identify overloaded students** - find students who registered for too many courses (more than 3)

How the program works:

- It stores student names: Alice, Bob, Charlie, Diana
- Each student has a list of courses they registered for
- Some students have fewer courses, some have more
- It can search through all students' courses to find who is taking a particular subject
- It counts how many courses each student has and flags those with too many

Imagine you're a college advisor. Students come to you with their course selection forms. You need to check who's taking which courses and make sure no student is taking too many classes at once.

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```c
#include <stdio.h>
#include <string.h>

int main()
{
    char students[4][20] = {"Alice", "Bob", "Charlie", "Diana"};
    char courses[4][10][20] = {
        {"Math", "Physics", "English"},
        {"Biology", "Chemistry"},
        {"Math", "Physics", "Computer", "History"},
        {"Math", "English", "Physics", "Chemistry", "Art"}
    };
    int courseCount[4] = {3, 2, 4, 5}, i, j;
    printf("Student Course Registration:\n\n");
    for (i = 0; i < 4; i++)
    {
        printf("%s is registered for: ", students[i]);
        for (j = 0; j < courseCount[i]; j++)
        {
            printf("%s", courses[i][j]);
            if (j < courseCount[i] - 1) printf(", ");
        }
        printf("\n");
    }
    char searchCourse[20] = "Physics";
    printf("\nStudents taking %s:\n", searchCourse);
    int found = 0;
    for (i = 0; i < 4; i++)
    {
        for (j = 0; j < courseCount[i]; j++)
        {
            if (strcmp(courses[i][j], searchCourse) == 0)
            {
                printf("- %s\n", students[i]);
                found = 1;
                break;
            }
        }
    }
    if (!found) printf("No students are taking %s.\n", searchCourse);
    printf("\nOverloaded Students (more than 3 courses):\n");
    int overloadFound = 0;
    for (i = 0; i < 4; i++)
    {
        if (courseCount[i] > 3)
        {
            printf("- %s (%d courses)\n", students[i], courseCount[i]);
            overloadFound = 1;
        }
    }
    if (!overloadFound) printf("No students are overloaded.\n");
    return 0;
}
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
Student Course Registration:

Alice is registered for: Math, Physics, English
Bob is registered for: Biology, Chemistry
Charlie is registered for: Math, Physics, Computer, History
Diana is registered for: Math, English, Physics, Chemistry, Art

Students taking Physics:
- Alice
- Charlie
- Diana

Overloaded Students (more than 3 courses):
- Charlie (4 courses)
- Diana (5 courses)

--------------------------------
Process exited after 0.06486 seconds with return value 0
Press any key to continue . . .
```

Question 8:

You own a restaurant and want to create a digital menu for your customers. Your menu has 3 categories (Appetizers, Main Course, Desserts) with 3 items in each category. The program should:

1. Display the complete menu organized by categories
2. Show budget-friendly options - find all items that cost less than $10
3. Format everything neatly so it's easy for customers to read

How the program works:

- It stores menu categories: Appetizers, Main Course, Desserts
- Each category has food items with prices
- It displays the menu in sections with clear headings
- It goes through all prices to find items that cost less than $10
- It shows which category each budget item belongs to

Imagine you're looking at a restaurant menu. The menu is divided into sections (starters, main dishes, desserts). You might look for the cheaper items if you're on a budget.

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```c
#include <stdio.h>

int main()
{
    char categories[3][20] = {
        "Appetizers",
        "Main Course",
        "Desserts"
    };
    char items[3][3][30] = {
        {"Spring Rolls", "Garlic Bread", "Soup"},
        {"Grilled Chicken", "Pasta Alfredo", "Beef Burger"},
        {"Ice Cream", "Chocolate Cake", "Fruit Salad"}
    };
    float prices[3][3] = {
        {6.50, 5.00, 7.25},
        {14.00, 12.50, 10.00},
        {4.50, 6.75, 5.50}
    };
    int i, j;
    printf("RESTAURANT MENU\n\n");
    for (i = 0; i < 3; i++)
    {
        printf("%s:\n", categories[i]);
        printf("--------------------------\n");

        for (j = 0; j < 3; j++)
        {
            printf("%-20s $%.2f\n", items[i][j], prices[i][j]);
        }
        printf("\n");
    }
    printf("BUDGET-FRIENDLY OPTIONS (< $10)=\n");
    int found = 0;
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++)
        {
            if (prices[i][j] < 10)
            {
                printf("%-20s $%.2f    (%s)\n",
                       items[i][j], prices[i][j], categories[i]);
                found = 1;
            }
        }
    }
    if (!found)
        printf("No budget-friendly options available.\n");
    return 0;
}
```

Student Name: _____Wajiha Nazir_____Roll No: _____25K-3021_____ Section: BSE-1B

```
RESTAURANT MENU

Appetizers:
---------------------------
Spring Rolls          $6.50
Garlic Bread          $5.00
Soup                  $7.25

Main Course:
---------------------------
Grilled Chicken       $14.00
Pasta Alfredo         $12.50
Beef Burger           $10.00

Desserts:
---------------------------
Ice Cream             $4.50
Chocolate Cake        $6.75
Fruit Salad           $5.50

BUDGET-FRIENDLY OPTIONS (< $10)=
Spring Rolls          $6.50   (Appetizers)
Garlic Bread          $5.00   (Appetizers)
Soup                  $7.25   (Appetizers)
Ice Cream             $4.50   (Desserts)
Chocolate Cake        $6.75   (Desserts)
Fruit Salad           $5.50   (Desserts)

--------------------------------
Process exited after 0.09572 seconds with return value 0
```