NAME: JITHENDRA BOODATI

REG NO: 22BCE1947

PAT 1

Write a C program to check whether a number is prime, armstrong, perfect number or not using functions.

Input:

11

Output:

11 is prime number

11 is not a armstrong number

11 is not a perfect number

CODE:

```
#include<stdio.h>

int is_prime(int n);
int is_armstrong(int n);
int is_perfect(int n);

int main()
{
   int n;
   printf("Enter a number: ");
   scanf("%d", &n);

   if(is_prime(n))
```

```
        printf("%d is prime number\n", n);
    else
        printf("%d is not a prime number\n", n);

    if(is_armstrong(n))
        printf("%d is a armstrong number\n", n);
    else
        printf("%d is not a armstrong number\n", n);

    if(is_perfect(n))
        printf("%d is a perfect number\n", n);
    else
        printf("%d is not a perfect number\n", n);

    return 0;
}

int is_prime(int n)
{
    int i;
    for(i=2; i<=n/2; i++)
    {
        if(n%i == 0)
            return 0;   // not a prime number
    }
    return 1;   // prime number
}

int is_armstrong(int n)
```

```
{
    int sum=0, rem, temp=n, digits=0;
    while(temp > 0)
    {
        digits++;
        temp /= 10;
    }
    temp = n;
    while(temp > 0)
    {
        rem = temp%10;
        sum += pow(rem, digits);
        temp /= 10;
    }
    if(n == sum)
        return 1;   // armstrong number
    else
        return 0;   // not an armstrong number
}

int is_perfect(int n)
{
    int i, sum=0;
    for(i=1; i<n; i++)
    {
        if(n%i == 0)
            sum += i;
    }
    if(n == sum)
```

```
        return 1;   // perfect number
    else
        return 0;   // not a perfect number
}
```

PAT 2

Write a c program to find the number of words,vowels, consonants, space and special characters in a string

INPUT:

*Nothing is impossible in this world.

OUTPUT:

```
Words = 6
Vowels = 10
Consonants = 20
Space = 5
Special Characters = 2
```

CODE:

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

int main()
{
    char str[100];
    int i, words=1, vowels=0, consonants=0, spaces=0, special=0;
    printf("Enter a string: ");
```

```
gets(str);

for(i=0; str[i]!='\0'; i++)
{
    if(str[i] == ' ')
    {
        words++;
        spaces++;
    }
    else if(str[i]>='a' && str[i]<='z' || str[i]>='A' && str[i]<='Z')
    {
        if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u' ||
           str[i]=='A' || str[i]=='E' || str[i]=='I' || str[i]=='O' || str[i]=='U')
        {
            vowels++;
        }
        else
        {
            consonants++;
        }
    }
    else if(str[i]>='0' && str[i]<='9')
    {
        // ignore digits
    }
    else
```

```
        {
          special++;
        }
    }
    printf("\nWords = %d\n", words);
    printf("Vowels = %d\n", vowels);
    printf("Consonants = %d\n", consonants);
    printf("Space = %d\n", spaces);
    printf("Special Characters = %d\n", special);

    return 0;
}
```

CAT 1a

Write a C program that accepts a string as input,print the length of the string and display the word fequency, then use pointers to find the first repeated and non-repeated character in the string, and print the output:

POSSIBLE TEST CASES:

INPUT:

SUJITHRA

OUTPUT:

```
Length of the string is: 8
Word frequency is: 8
No repeated characters found in the string.
First non-repeated character is: S
```

#2 INPUT:

ASSDFG

OUTPUT:

```
Length of the string is: 6
Word frequency is: 5
First repeated character is: S
First non-repeated character is:
```

#3 INPUT:

```
RUDRESH
```

OUTPUT:

```
Length of the string is: 7
Word frequency is: 6
First repeated character is: R
First non-repeated character is: U
```

CODE:

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

void print_output(char *str, int len);

int main()
{
    char str[100];
    int len;
    printf("Enter a string: ");
    gets(str);
    len = strlen(str);
    print_output(str, len);


    return 0;
}


void print_output(char *str, int len)
{
    int i, j, word_freq = 1;
    char *first_repeat = NULL, *first_non_repeat = NULL;
    int freq_arr[26] = {0};


    for (i = 0; i < len; i++)
    {
        if (str[i] == ' ')
        {
            word_freq++;
        }
        else
        {
            freq_arr[str[i] - 'a']++;
            if (first_repeat == NULL && freq_arr[str[i] - 'a'] == 2)
            {
                first_repeat = &str[i];
            }
            else if (first_non_repeat == NULL && freq_arr[str[i] - 'a'] == 1)
            {
                first_non_repeat = &str[i];
            }
        }
    }

    printf("Length of the string is: %d\n", len);
    printf("Word frequency is: %d\n", word_freq);

    if (first_repeat != NULL)
    {
        printf("First repeated character is: %c\n", *first_repeat);
    }
    else
    {
        printf("No repeated characters found in the string.\n");
    }

    if (first_non_repeat != NULL)
    {
        printf("First non-repeated character is: %c\n", *first_non_repeat);
    }
    else
    {
        printf("No non-repeated characters found in the string.\n");
    }
}
```

<u>CAT 1b</u>

Write a cprogram to get the employee information name,age,position and Date of joining. Print the employee list based on Alphabaetical order. Display the order of the employees based on date of joining.

SAMPLE INPUT MODEL:

```
Enter the number of employees: 3
Enter details of employee 1:
Name: Jane
Age: 34
Position: HR
Date of joining (dd/mm/yyyy): 10/2/2000
Enter details of employee 2:
Name: Amie
Age: 23
Position: Sales
Date of joining (dd/mm/yyyy): 12/03/2004
Enter details of employee 3:
Name: Balu
Age: 45
Position: Security
Date of joining (dd/mm/yyyy): 1/1/1998
```

SAMPLE OUTPUT MODEL:

```
Employee List sorted by name:
Name: Amie
Age: 23
```

```
Position: Sales

Date of Joining: 12/03/2004

Name: Balu

Age: 45

Position: Security

Date of Joining: 1/1/199

Name: Jane

Age: 34

Position: HR

Date of Joining: 10/2/200

Employee List sorted by date of joining:

Name: Balu

Age: 45

Position: Security

Date of Joining: 1/1/1998

Name: Jane

Age: 34

Position: HR

Date of Joining: 10/2/2000

Name: Amie

Age: 23

Position: Sales

Date of Joining: 12/03/2004
```

PUBLIC TEST CASE:

```
3

Jane
```

34

HR

10/2/2000

Amie

23

Sales

12/03/2004

Balu

45

Security

1/1/1998

OUTPUT:

```
Employee List sorted by name:

Amie
23
Sales
12/03/2004
Balu
45
Security
1/1/1998
Jane
34
HR
10/2/2000
```

```
Employee List sorted by date of joining:
Balu
Jane
Amie
```

CODE:

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

struct Employee {
    char name[100];
    int age;
    char position[100];
    char date_of_joining[11];
};

void print_employee_list_name_sort(struct Employee *emp, int n) {
    // Sort employees based on name
    for(int i=0; i<n-1; i++) {
        for(int j=i+1; j<n; j++) {
            if(strcmp(emp[i].name, emp[j].name) > 0) {
                struct Employee temp = emp[i];
                emp[i] = emp[j];
                emp[j] = temp;
            }
        }
    }
```

```
// Print employee list sorted by name
printf("\nEmployee List sorted by name:\n\n");
for(int i=0; i<n; i++) {
    printf("Name: %s\n", emp[i].name);
    printf("Age: %d\n", emp[i].age);
    printf("Position: %s\n", emp[i].position);
    printf("Date of Joining: %s\n\n", emp[i].date_of_joining);
```