ASSIGNMENT -2

Q] WACP to find the nature of roots of a second order quadratic equation. Also compute the roots of the equation.

ALGORITHM:--

- ***Start.***
- ***Input a, b, c as well as input discriminant,root1,root2,realPart,imgPart.***
- ***discriminant ← (b × b – 4 × a × c).***
- ***if(discriminant>0) then print root1 ← (-b + d) / (2 × a) and root2 ← (-b - d) / (2 × a).***
- ***if(discriminant == 0) then print root1=root2=-b/(2*a).***
- ***else print realPart and imgPart.***
- ***Stop.***

main.c

Run

```c
#include <math.h>
#include <stdio.h>
int main() {
   int a,b,c;
   float discriminant, root1, root2, realPart, imagPart;
    printf("Enter coefficients a, b and c: ");
    scanf("%d %d %d", &a, &b, &c);

    discriminant = b * b - 4 * a * c;

    if (discriminant > 0) {
        root1 = (-b + sqrt(discriminant)) / (2 * a);
        root2 = (-b - sqrt(discriminant)) / (2 * a);
        printf("root1 = %.2f and root2 = %.2f", root1, root2);
    }
    else if (discriminant == 0) {
        root1 = root2 = -b / (2 * a);
        printf("root1 = root2 = %.2f;", root1);
    }
    else {
        realPart = -b / (2 * a);
        imagPart = sqrt(-discriminant) / (2 * a);
        printf("root1 = %.2f+%.2fi and root2 = %.2f-%.2fi", realPart, imagPart
            , realPart, imagPart);
    }

    return 0;
}
```

Output

Clear

```
/tmp/2czKqQ86Z5.o
Enter coefficients a, b and c: 4 5 6
root1 = 0.00+1.05i and root2 = 0.00-1.05i
```

Q] WACP to check whether the user given year is a Leap Year or not and display the result on screen.

ALGORITHM:--

1] START

2] Step 1 → Take integer variable year

3] Step 2 → Assign value to the variable

4] Step 3 → Check if year is divisible by 4 and 400, DISPLAY "is a leap year"

5] Step 4 → Check if year is divisible by 100, DISPLAY "is not a leap year"

6] Step 5 → Otherwise, DISPLAY " is not leap year"

7] STOP

main.c

Run

```c
#include <stdio.h>
int main() {
    int year;
    printf("Enter a year: ");
    scanf("%d", &year);

    if (year % 400 == 0) {
        printf("%d is a leap year.", year);
    }
    else if (year % 100 == 0) {
        printf("%d is not a leap year.", year);
    }

    else if (year % 4 == 0) {
        printf("%d is a leap year.", year);
    }
    else {
        printf("%d is not a leap year.", year);
    }

    return 0;
}
```

Output

Clear

```
/tmp/7i4qZfReBP.o
Enter a year: 2004
2004 is a leap year.
```

Q] WACP to find the greatest among the three user given numbers using (a) if-else- if statement, (b) using ternary operator. and display the result on screen.

ALGORITHM:--

1] Start

2] Take Three Doubles Variables ie n1, n2 , n3.

3] Check if n1 > n2 and n1>n3 then print n1.

4] Check if n2>n3 and n2>n3 then print n2.

5] If the above Conditions does not satisfies then print n3.

6] Stop.

main.c

Run

```c
#include <stdio.h>
int main() {
    double n1, n2, n3;
    printf("Enter three numbers: ");
    scanf("%lf %lf %lf", &n1, &n2, &n3);


    if (n1 >= n2 && n1 >= n3)
        printf("%.2lf is the largest number.", n1);


    else if (n2 >= n1 && n2 >= n3)
        printf("%.2lf is the largest number.", n2);


    else
        printf("%.2lf is the largest number.", n3);

    return 0;
}
```

Output

Clear

```
/tmp/2tKPjq4Uhr.o
Enter three numbers: 54 69 12
69.00 is the largest number.
```

Q] WACP to check whether a number is: (a) Prime

ALGORITHM:--

Step 1: Start
Step 2: Initialise variables n , flag=1, i=2;
Step 3: Read n from user
Step 4: If n<=1 \
Display "n is not a prime number"
Go to step 7
Step 5: Repeat the steps until i<[(n/2)+1]
5.1) If remainder of number divide j equals to 0,
Set flag=0
Go to step 6
5.2) i=i+1
Step 6: If flag==0,
Display n " is not prime number";
Else
Display n " n is prime number";

Step 7: Stop

main.c

Run

```c
#include <stdio.h>
int main() {
  int n, i, flag = 0;
  printf("Enter a positive integer: ");
  scanf("%d", &n);

  for (i = 2; i <= n / 2; ++i) {
    if (n % i == 0) {
      flag = 1;
      break;
    }
  }
if (n == 1) {
    printf("1 is neither prime nor composite.");
  }
  else {
    if (flag == 0)
      printf("%d is a prime number.", n);
    else
      printf("%d is not a prime number.", n);
  }

  return 0;
}
```

Output

Clear

```
/tmp/lmXmgTLKnA.o
Enter a positive integer: 5
is a prime number.
```

Q] WACP to check whether a number is: (b)Armstrong.

ALGORITHM:--

Step I: Start.

Step II: Input num, originalsum, remainder, result;

Step III: sum = 0, rem=0.

Step IV: Print “Enter three-digited integer : ”

Step V: Read num.

Step VI: orginalsum=num;

Step VII: If originalsum not equal to zero Then,

remainder = originalsum mod 10

result += (rem X rem X rem);

originalsum/=10;

Step VIII: If (result is equal to num) Then,

Print “Number num is an Armstrong number.”

Else

Print “Number num is not an Armstrong number.”

Step IX: Stop

main.c

```c
#include <stdio.h>
int main() {
    int num, originalNum, remainder, result = 0;
    printf("Enter a three-digit integer: ");
    scanf("%d", &num);
    originalNum = num;

    while (originalNum != 0) {

        remainder = originalNum % 10;

       result += remainder * remainder * remainder;
       originalNum /= 10;
    }

    if (result == num)
        printf("%d is an Armstrong number.", num);
    else
        printf("%d is not an Armstrong number.", num);

    return 0;
}
```

Output

```
/tmp/XBBZeU4Pin.o
Enter a three-digit integer: 370
370 is an Armstrong number.
```

THANK YOU