Lab Assingment-3

ROLL: 2005535 | NAME: SAHIL SINGH | DATE: 12/08/21

QUES 1: WAP to take input as a string and convert into its equivalent number and return the same.

SOLUTION:

```
#include <iostream>
using namespace std;
int main()
{
    char str[50];

    cout << "Enter a string : ";
    gets(str);

    for (int i = 0; str[i] != '\0'; i++)
    {
        cout << "ASCII value of " << str[i] << " is " << int(str[i]) << endl;
    }
    return 0;
}
```

OUTPUT:

```
Enter a string : Sahil Singh
ASCII value of S is 83
ASCII value of a is 97
ASCII value of h is 104
ASCII value of i is 105
ASCII value of l is 108
ASCII value of   is 32
ASCII value of S is 83
ASCII value of i is 105
ASCII value of n is 110
ASCII value of g is 103
ASCII value of h is 104
```

QUES 2: Create a structure Point with X and Y coordinates and perform the following operations:- a) Given two end points of a line segment, find its length. b) Given the centre point and radius of a circle, find if point P lies inside the circle. c) Given two line segments find if they are parallel or not. d) Given three vertices of a triangle, check if it is a valid triangle, then find its area.

SOLUTION:

```
#include <iostream>
#include <cmath>
using namespace std;
struct Point
```

```
{
    float x;
    float y;
};
float length_of_line(Point, Point);
bool is_inside_circle(Point, float, Point);
bool is_parallel(Point, Point, Point, Point);
float area_of_triangle(Point, Point, Point);
void input(Point &);
int main()
{
    int radius, choice;
    Point a, b, c, d;
    do
    {
        cout << "Enter Your Choice:\n1) Length of a Line Segment\n";
        cout << "2) Is point inside the circle?\n3) Are the lines Parallel?\n";
        cout << "4) What is the area of the triangle?\n5) Exit\n->:";
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter Point 1: ";
            input(a);
            cout << "Enter Point 2: ";
            input(b);
            cout << "=Length: " << length_of_line(a, b) << endl;
            break;
        case 2:
            cout << "Enter circle's centre point: ";
            input(a);
            cout << "Enter circle's radius: ";
            cin >> radius;
            cout << "Enter the point: ";
            input(b);
            cout << "=The point is ";
            cout << (is_inside_circle(a, radius, b) ? "Inside\n" : "Outside\n");
            break;
        case 3:
            cout << "Line 1:----\nPoint 1: ";
            input(a);
            cout << "Point 2: ";
            input(b);
            cout << "Line 2:----\nPoint 1: ";
            input(c);
            cout << "Point 2: ";
            input(d);
            cout << "=The lines are ";
            cout << (is_parallel(a, b, c, d) ? "parallel\n" : "Not Parallel\n");
            break;
        case 4:
            cout << "Enter Vertex 1: ";
            input(a);
            cout << "Enter Vertex 2: ";
            input(b);
            cout << "Enter Vertex 3: ";
            input(c);
            if (area_of_triangle(a, b, c))
            {
                cout << "=Area: " << area_of_triangle(a, b, c) << endl;
                break;
            }
            cout << "=The triange is not possible!\n";
            break;
        default:
            cout << "\nExiting...\n";
        }
        cout << "--------------------" << endl;
    } while (choice >= 1 && choice <= 4);
    return 0;
}
float length_of_line(Point a, Point b)
{
    return sqrt(pow((a.x - b.x), 2) + pow((a.y - b.y), 2));
}
bool is_inside_circle(Point centre, float radius, Point P)
{
    float dist = length_of_line(centre, P);
    if (dist <= radius)
        return true;
    return false;
}
bool is_parallel(Point a, Point b, Point p, Point q)
{
    float m1 = (b.y - a.y) / (b.x - a.x);
    float m2 = (q.y - p.y) / (q.x - p.x);
    if (m1 == m2)
        return true;
    return false;
}
float area_of_triangle(Point a, Point b, Point c)
{
    float area = (b.x * c.y - c.x * b.y) - (a.x * c.y - c.x * a.y) + (a.x * b.y - b.x * a.y);
    return abs(area / 2);
}
void input(Point &a) { cin >> a.x >> a.y; }
```

OUTPUT:

```
Enter Your Choice:
```

```
1) Length of a Line Segment
2) Is point inside the circle?
3) Are the lines Parallel?
4) What is the area of the triangle?
5) Exit
->:1
Enter Point 1: 2 5
Enter Point 2: 3 4
= Length: 1.41421
--------------------
Enter Your Choice:
1) Length of a Line Segment
2) Is point inside the circle?
3) Are the lines Parallel?
4) What is the area of the triangle?
5) Exit
->:5

Exiting...
--------------------
```

QUES 3: WAP that calculates area and perimeter of the following geometric figures. Your function should use function overloading and return both area and perimeter. a) Square b) Circle c) Rectangle d) Triangle

SOLUTION:

```cpp
#include <iostream>
#include <cmath>
using namespace std;
struct Details
{
    float area;
    float perimeter;
};
Details area_perimeter(float);
Details area_perimeter(double);
Details area_perimeter(float, float);
Details area_perimeter(float, float, float);
int main()
{
    Details square, circle, rectangle, triangle;
    int choice;
    do
    {
        cout << "Enter your choice:\n1) Square\n2) Circle\n";
        cout << "3) Rectangle\n4) Triangle\n5) Exit\n->:";
        cin >> choice;
        switch (choice)
        {
```

```
        case 1:
            float a;
            cout << "Enter a side of square: ";
            cin >> a;
            square = area_perimeter(a);
            cout << "=Area: " << square.area << endl;
            cout << "=perimeter: " << square.perimeter << endl;
            break;
        case 2:
            double r;
            cout << "Enter radius of circle: ";
            cin >> r;
            circle = area_perimeter(r);
            cout << "=Area: " << circle.area << endl;
            cout << "=perimeter: " << circle.perimeter << endl;
            break;
        case 3:
            float l, b;
            cout << "Enter Length and breadth of rectangle: ";
            cin >> l >> b;
            rectangle = area_perimeter(l, b);
            cout << "=Area: " << rectangle.area << endl;
            cout << "=perimeter: " << rectangle.perimeter << endl;
            break;
        case 4:
            float side_1, side_2, side_3;
            cout << "[Make sure (a+b)>c]\n";
            cout << "Enter sides of the triangle: ";
            cin >> side_1 >> side_2 >> side_3;
            triangle = area_perimeter(side_1, side_2, side_3);
            cout << "=Area: " << triangle.area << endl;
            cout << "=perimeter: " << triangle.perimeter << endl;
            break;
        default:
            cout << "\nExiting...\n";
        }
        cout << "-----------------\n";
    } while (choice >= 1 && choice <= 4);
    return 0;
}
Details area_perimeter(float a)
{
    Details detail;
    detail.area = a * a;
    detail.perimeter = a * 4;
    return detail;
}
Details area_perimeter(double r)
{
    Details detail;
    detail.area = 3.14159 * r * r;
    detail.perimeter = 2 * 3.14159 * r;
    return detail;
}
Details area_perimeter(float l, float w)
{
    Details detail;
    detail.area = l * w;
    detail.perimeter = 2 * (l + w);
    return detail;
}
Details area_perimeter(float a, float b, float c)
{
    Details detail;
    float s = (a + b + c) / 2;
    detail.area = sqrt(s * (s - a) * (s - b) * (s - c));
    detail.perimeter = a + b + c;
    return detail;
}
```

OUTPUT:

```
Enter your choice:
1) Square
2) Circle
3) Rectangle
4) Triangle
5) Exit
->:1
Enter a side of square: 5
=Area: 25
=perimeter: 20
-----------------
Enter your choice:
1) Square
2) Circle
3) Rectangle
4) Triangle
5) Exit
->:2
Enter radius of circle: 5
=Area: 78.5397
=perimeter: 31.4159
-----------------
Enter your choice:
1) Square
2) Circle
3) Rectangle
4) Triangle
5) Exit
```

```
->:5

Exiting...
-----------------
```

QUES 4: Write a function myAvg() that takes undefined number of inputs and returns the average of the same.

SOLUTION:

```cpp
#include <iostream>
#define INPUT_MAX 100000
using namespace std;
double myAvg(double input[], int i);
int main()
{
    double input[INPUT_MAX];
    int i = 0;
    cout << "*Enter any character to stop input*\n\n";
    while (cin >> input[i])
    {
        i++;
    }
    float average = (float)myAvg(input, i);
    cout << "Average: " << average << endl;
    return 0;
}
double myAvg(double input[], int i)
{
    double sum = 0;
    int count = 0;
    while (count != i)
    {
        sum += input[count];
        count++;
    }
    return sum / count;
}
```

OUTPUT:

```
*Enter any character to stop input*

14 25 56 34 45.33 56 67 56 A
Average: 44.1662
```

QUES 5: Write a function myAvg() that takes undefined number of inputs and returns the average of the same.

OUTPUT:

```
#include <iostream>
#include <stdarg.h>
using namespace std;
double myAvg(int num, ...);
int main()
{
    float average = (float)myAvg(3, 1.0, 2.0, 3.0);
    cout << "Average: " << average << endl;
    average = (float)myAvg(5, 2.6, 2.7, 15.1, 12.7, 19.9);
    cout << "Average: " << average << endl;
    return 0;
}
double myAvg(int num, ...)
{
    va_list valist;
    double sum = 0;
    va_start(valist, num);
    for (int i = 0; i < num; i++)
        sum += va_arg(valist, double);
    va_end(valist);
    return sum / num;
}
```

OUTPUT:

```
Average: 2
Average: 10.6
```

QUES 6: Write a program that takes input string (student names) as command line arguments and outputs a greeting massage to all of them.

SOLUTION:

```
#include <iostream>
using namespace std;
void print_hello(char *arg[], int a);
int main(int a, char *arg[])
{
    print_hello(arg, a);
    return 0;
}
void print_hello(char *arg[], int a)
{
    if (a == 1)
        return;
    cout << "Hello ";
    for (int i = 1; i < a - 1; i++)
        cout << arg[i] << ", ";
    if (a >= 3)
        cout << "\b\b and " << *(arg + a - 1);
    else if (a == 2)
        cout << *(arg + 1);
    cout << ", Welcome to KIIT\n";
}
```

OUTPUT:

```
PS D:\oop_lab\week_3> g++ ques6.cpp
PS D:\oop_lab\week_3> ./a.exe Sahil AlQaeda Atankwadi
Hello Sahil, AlQaeda and Atankwadi, Welcome to KIIT
```

QUES 7: WAP with recursive functions to perform the following:- a) Reverse an input string. b) Check if an input string is palindrome or not. c) Binary Search. d) Sorting by putting the least number in beginning with each successive call.

SOLUTION:

```
#include <iostream>
#include <string.h>
using namespace std;
void rev_string(char[], int, int left = 0);
bool isPalindrome(char[], int, int left = 0);
int binary_search(char[], char, int, int left = 0);
void selection_sort(char[], int, int left = 0);
int main()
{
    char str[50], ch;
    int choice;
    do
    {
        cout << "Enter a choice: \t(string: " << str << ")\n";
        cout << "1)Enter new string\n2)Reverse the string\n3)Check if palindrome\n";
        cout << "4)Do a binary search\n5)Sort the string\n6)Exit\n::";
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter String: ";
            cin >> str;
            break;
        case 2:
            cout << "Original String: " << str << endl;
            rev_string(str, strlen(str));
            cout << "Reversed String: " << str << endl;
            break;
        case 3:
            cout << "\nThe string is palindrome: ";
            cout << (isPalindrome(str, strlen(str)) ? "true" : "false") << endl;
            break;
        case 4:
```

```cpp
            cout << "Enter the element to search for: ";
            cin >> ch;
            cout << "The element is at index: ";
            cout << binary_search(str, ch, strlen(str)) << endl;
            break;
        case 5:
            cout << "Original String: " << str << endl;
            selection_sort(str, strlen(str));
            cout << "Sorted String: " << str << endl;
            break;
        default:
            cout << "\nExiting...\n";
        }
        cout << "--------------" << endl;
    } while (choice >= 1 && choice <= 5);
    return 0;
}
void rev_string(char str[], int right, int left)
{
    if (left - 1 < right)
    {
        swap(str[left - 1], str[right]);
        rev_string(str, right - 1, left + 1);
    }
}
bool isPalindrome(char str[], int right, int left)
{
    if (str[left] != str[right - 1])
        return false;
    else if (left < right - 1)
        isPalindrome(str, right - 1, left + 1);
    return true;
}
int binary_search(char str[], char ch, int right, int left)
{
    int mid = (left + right - 1) / 2;
    if (right > left)
    {
        if (str[mid] == ch)
            return mid;
        else if (str[mid] < ch)
            return binary_search(str, ch, right, mid + 1);
        else
            return binary_search(str, ch, mid - 1, left);
    }
    return -1;
}
void selection_sort(char str[], int n, int left)
{
    if (left == n)
```

```
        return;
    int min = left;
    for (int i = left + 1; i < n; i++)
        if (str[min] > str[i])
            min = i;
    swap(str[left], str[min]);
    selection_sort(str, n, left + 1);
}
```

OUTPUT:

```
Enter a choice:         (string: ≥u+¥K)
1)Enter new string
2)Reverse the string
3)Check if palindrome
4)Do a binary search
5)Sort the string
6)Exit
::1
Enter String: Sahil
--------------
Enter a choice:         (string: Sahil)
1)Enter new string
2)Reverse the string
3)Check if palindrome
4)Do a binary search
5)Sort the string
6)Exit
::2
Original String: Sahil
Reversed String: lihaS
--------------
Enter a choice:         (string: lihaS)
1)Enter new string
2)Reverse the string
3)Check if palindrome
4)Do a binary search
5)Sort the string
6)Exit
::3

The string is palindrome: false
--------------
Enter a choice:         (string: lihaS)
1)Enter new string
2)Reverse the string
3)Check if palindrome
4)Do a binary search
5)Sort the string
6)Exit
```

```
::6

Exiting...
--------------
```