Date : 02.02.2025
Lakshana Baskaran
24011103026

LAB - 2
Pointers and Classes

QUESTION 1:

Write a C++ menu-driven program to determine whether a number is a Palindrome, Armstrong, or Perfect Number. Normal variable and array declarations are not allowed. Utilize dynamic memory allocation (DMA). Design proper functions, maintain boundary conditions, and follow coding best practices. The menu is as follows:

a. Palindrome
b. Armstrong Number
c. Perfect Number
d. Exit

SOURCE CODE:

```
#include <iostream>
#include <stdbool.h>
using namespace std;

int* reverse(int*);
bool palindrome(int*);

int* len(int*);
int* pow(int*, int*);
bool armstrong_number(int*);
bool perfect_number(int*);

//returns the reverse of an integer
int* reverse(int *m) {
    int *u = (int*) malloc (sizeof(int));
    int *r = (int*) malloc (sizeof(int));
    int *n = (int*) malloc (sizeof(int));
    *n = *m;

    *r = 0;
```

```
    while ((*n) > 0) {
        *u = (*n) % 10;
        *r = (*r)*10 + (*u);
        *n = (*n) / 10;
    }

    free(u);
    return r;
}

//checks if the integer is palindrome or not
bool palindrome(int* n) {
    int *u = (int*) malloc (sizeof(int));
    *u = *n;

    if (*u == (*reverse(n))) {
        free(u);
        return true;
    }

    free(u);
    return false;
}

//returns the length of the integer
int* len(int *n) {
    int *u = (int*) malloc (sizeof(int));
    *u = *n;
    int *count = (int*) malloc (sizeof(int));
    *count = 0;
    while ((*u) > 0) {
        (*u) = (*u) /10;
        (*count)++;
    }

    return count;
}

//returns a^m
int* pow(int* a, int* m) {
    int *e = (int*) malloc (sizeof(int));
    *e = 1;
```

```
    int *b = (int*) malloc (sizeof(int));
    *b = *m;
    for(; (*b) > 0; (*b)--) {
        *e = (*e) * (*a);
    }

    return e;
}

//checks if the integer is an armstrong number or not
bool armstrong_number(int *n) {
    int *l = (int*) malloc (sizeof(int));
    *l = *(len(n));

    int *arm = (int*) malloc (sizeof(int));
    *arm = 0;

    int *m = (int*) malloc (sizeof(int));
    *m = *n;

    int *u = (int*) malloc (sizeof(int));

    while ((*m) > 0) {
        *u = (*m) % 10;
        *arm = (*arm) + (*pow(u, l));
        *m = (*m) / 10;
    }
    free(u);
    free(l);
    free(m);

    if ((*arm) == (*n)) {
        free(arm);
        return true;
    }
    free(arm);
    return false;
}

//checks if the integer is a perfect number or not
bool perfect_number(int* n) {
    int *p = (int*) malloc (sizeof(int));
    *p = 0;
```

```
    int *i = (int*) malloc (sizeof(int));

    for ((*i) = 1; (*i) < (*n); (*i)++) {
        if ((*n) % (*i) == 0) {
            (*p) = (*p) + (*i);
        }
    }
    if ((*p) == (*n)){
        return true;
    }
    return false;
    }

int main() {
    int *n = (int*) malloc (sizeof(int));

    printf("Enter a number: ");
    scanf("%d", n);

    int *choice = (int*) malloc (sizeof(int));
    printf("\n1 - Palindrome or Not\n2 - Armstrong Number or Not\n3 -
Perfect Number or Not\n4 - Exit");
    while ((*choice) != 4) { //menu
        printf("\nENter you choice:");
        scanf("%d", choice);

        switch (*choice) {
            case 1:
                if (palindrome(n) == true)
                    cout << "IT IS A PALINDROME" << endl;
                else
                    cout << "IT IS NOT A PALINDROME" << endl;
                break;

            case 2:
                if (armstrong_number(n) == true)
                    cout << "IT IS AN ARMSTRONG NUMBER!!!" << endl;
                else
                    cout << "IT IS NOT AN ARMSTRONG NUMBER." << endl;
                break;

            case 3:
                if (perfect_number(n) == true)
```

Date : 02.02.2025
Lakshana Baskaran
24011103026

```
                    cout << "IT IS AN PERFECT NUMBER!!!" << endl;
                else
                    cout << "IT IS NOT AN PERFECT NUMBER." << endl;
                break;

            case 4:
                printf("exiting...\n");
                break;

            default:
                break;
        }
    }
    free(n);
    return 0;
}
```

OUTPUT:

```
lemon@jupiter:~/workspace/college/DSA/Lab-2$ g++ -o out pointer.cpp
lemon@jupiter:~/workspace/college/DSA/Lab-2$ ./out
Enter a number: 28

1 - Palindrome or Not
2 - Armstrong Number or Not
3 - Perfect Number or Not
4 - Exit
ENter you choice:1
IT IS NOT A PALINDROME

ENter you choice:2
IT IS NOT AN ARMSTRONG NUMBER.

ENter you choice:3
IT IS AN PERFECT NUMBER!!!

ENter you choice:4
exiting...
lemon@jupiter:~/workspace/college/DSA/Lab-2$
```

Date : 02.02.2025
Lakshana Baskaran
24011103026

QUESTION 2:

Write a C++ menu-driven program that calculates and displays the area of a square, cube, rectangle, and cuboid. Consider length as the side value for the square and cuboid. Identify proper data members and member functions. Design and create an appropriate class for the given scenario. Maintain proper boundary conditions and follow coding best practices. The menus are as follows:

a) Square
b) Cube
c) Rectangle
d) Cuboid
e) Exit

SOURCE CODE:

```
//menu driven program to calculate the area of square, cube,
rectangle, cuboid
#include <stdio.h>
#include <stdlib.h>
using namespace std;

//class for area
class area {
    private:
        int length, breadth, height;

    public:
        int square(int l);
        int cube(int l);
        int rectangle(int l, int b);
        int cuboid(int l, int b, int h);
};

//class method for area of square
int area::square(int l) {
    return l*l;
}

//class method for area of cube
int area::cube(int l) {
```

```
    return 6*l*l;
}

//class method for area of rectangle
int area::rectangle(int l, int b) {
    return l*b;
}

//class method for area of cuboid
int area::cuboid(int l, int b, int h) {
    return 2*(l*b + b*h + l*h);
}

int main(int argc, char* argv[]) {
    area obj;

    if (argc < 2) {
        return 0;
    }

    int *choice = (int*) malloc (sizeof(int));
    printf("\n1 - Square\n2 - Cube\n3 - Rectangle\n4 - Cuboid\n5 -
Exit\n");
    while ((*choice) != 5) { //menu
        printf("\nENter you choice:");
        scanf("%d", choice);

        switch (*choice) {
            case 1:
                printf("The area of the square of length %d is %d\n",
atoi(argv[1]), obj.square(atoi(argv[1])));
                break;
            case 2:
                printf("The surface area of the cube of edge length
%d is %d\n", atoi(argv[1]), obj.cube(atoi(argv[1])));
                break;
            case 3:
                printf("The area of the rectangle of length %d and
breadth %d is %d\n", atoi(argv[1]), atoi(argv[2]),
obj.rectangle(atoi(argv[1]), atoi(argv[2])));
                break;
            case 4:
```

```
                printf("The area of the cuboid of length %d, breadth
%d and height %d is %d\n", atoi(argv[1]), atoi(argv[2]),
atoi(argv[3]), obj.cuboid(atoi(argv[1]), atoi(argv[2]),
atoi(argv[3])));
                break;
            case 5:
                printf("exiting...\n");
                break;
            default:
                break;
        }
    }

    free(choice);

}
```

OUTPUT:

```
lemon@jupiter:~/workspace/college/DSA/Lab-2$ g++ -o out classes.cpp
lemon@jupiter:~/workspace/college/DSA/Lab-2$ ./out 1 2 3

  1 - Square
  2 - Cube
  3 - Rectangle
  4 - Cuboid
  5 - Exit

  ENter you choice:1
  The area of the square of length 1 is 1

  ENter you choice:2
  The surface area of the cube of edge length 1 is 6

  ENter you choice:3
  The area of the rectangle of length 1 and breadth 2 is 2

  ENter you choice:4
  The area of the cuboid of length 1, breadth 2 and height 3 is 22

  ENter you choice:5
  exiting...
lemon@jupiter:~/workspace/college/DSA/Lab-2$
```

Date : 02.02.2025
Lakshana Baskaran
24011103026