DAY 1

Gaurav Kumar 22BCS10159 KPIT 901-A

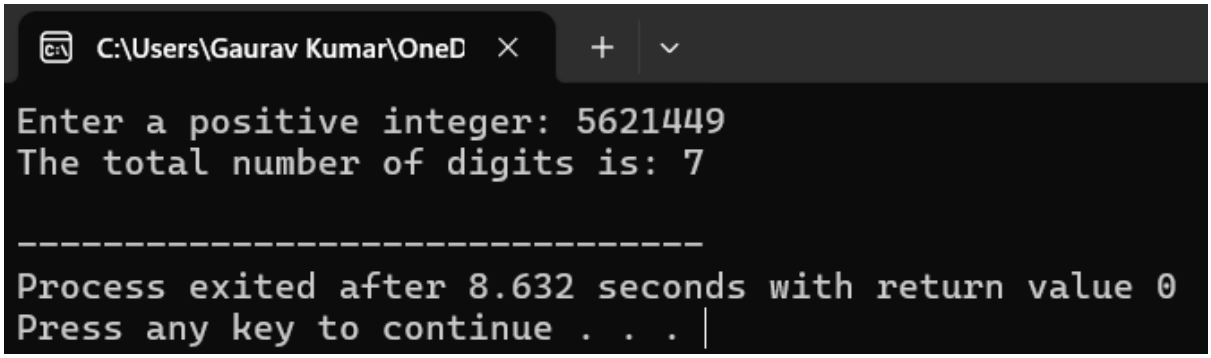
1) Sum of Natural Numbers up to N

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;
    if (n > 0) {
        int sum = n * (n + 1) / 2;
        cout << "Sum of natural numbers from 1 to " << n << " is: " << sum << endl;
    } else {
        cout << "Please enter a positive integer!" << endl;
    }
    return 0;
}
```

```
C:\Users\Gaurav Kumar\OneD
Enter a positive integer: 5
Sum of natural numbers from 1 to 5 is: 15

--------------------------------
Process exited after 9.454 seconds with return value 0
Press any key to continue . . .
```

1)Count Digits in a Number

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;

    if (n <= 0) {
        cout << "Please enter a positive integer!" << endl;
    } else {
        int digitCount = 0;
        while (n > 0) {
            n /= 10;
            digitCount++;
        }
        cout << "The total number of digits is: " << digitCount << endl;
    }
    return 0;
}
```



Implement Polymorphism for Banking Transactions

```cpp
#include <iostream>
#include <iomanip>
using namespace std;

class Account {
protected:
    double balance;
public:
    Account(double bal) : balance(bal) {}
    virtual void calculateInterest() = 0;
    virtual ~Account() {}
};

class SavingsAccount : public Account {
    double rate;
    int time;
public:
    SavingsAccount(double bal, double r, int t) : Account(bal), rate(r / 100), time(t) {}
    void calculateInterest() override {
        double interest = balance * rate * time;
        cout << fixed << setprecision(2)
             << "Savings Account Interest: " << interest
             << "\nFinal Balance: " << balance + interest << endl;
    }
};

class CurrentAccount : public Account {
    double maintenanceFee;
public:
    CurrentAccount(double bal, double fee) : Account(bal), maintenanceFee(fee) {}
    void calculateInterest() override {
        double deductions = maintenanceFee * 12;
        cout << fixed << setprecision(2)
             << "No Interest for Current Account.\nYearly Maintenance Deduction: "
             << deductions
             << "\nFinal Balance: " << balance - deductions << endl;
    }
};

int main() {
    cout << "Choose Account Type:\n1. Savings Account\n2. Current Account\n";
    int choice;
    cin >> choice;
```

```
44
45      if (choice == 1) {
46          double balance, rate;
47          int time;
48          cout << "Enter Balance: ";
49          cin >> balance;
50          cout << "Enter Interest Rate (in %): ";
51          cin >> rate;
52          cout << "Enter Time (in years): ";
53          cin >> time;
54
55          SavingsAccount sa(balance, rate, time);
56          sa.calculateInterest();
57      } else if (choice == 2) {
58          double balance, maintenanceFee;
59          cout << "Enter Balance: ";
60          cin >> balance;
61          cout << "Enter Monthly Maintenance Fee: ";
62          cin >> maintenanceFee;
63
64          CurrentAccount ca(balance, maintenanceFee);
65          ca.calculateInterest();
66      } else {
67          cout << "Invalid account type selected!" << endl;
68      }
69
70      return 0;
71  }
72
```

```
C:\Users\Gaurav Kumar\OneD
Choose Account Type:
1. Savings Account
2. Current Account
1
Enter Balance: 45623
Enter Interest Rate (in %): 5
Enter Time (in years): 3
Savings Account Interest: 6843.45
Final Balance: 52466.45

--------------------------------
Process exited after 65.52 seconds with return value 0
Press any key to continue . . .
```

```
C:\Users\Gaurav Kumar\OneD
Choose Account Type:
1. Savings Account
2. Current Account
2
Enter Balance: 45663254
Enter Monthly Maintenance Fee: 89
No Interest for Current Account.
Yearly Maintenance Deduction: 1068.00
Final Balance: 45662186.00

--------------------------------
Process exited after 11.99 seconds with return value 0
Press any key to continue . . .
```

Hierarchical Inheritance for Employee Management System

```cpp
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;

class Employee {
protected:
    string name;
    int id;
    double salary;
public:
    Employee(string empName, int empId, double empSalary)
        : name(empName), id(empId), salary(empSalary) {}

    virtual void calculateEarnings() = 0;
    virtual ~Employee() {}
};

class Manager : public Employee {
    int performanceRating;
public:
    Manager(string empName, int empId, double empSalary, int rating)
        : Employee(empName, empId, empSalary), performanceRating(rating) {}

    void calculateEarnings() override {
        double bonus = 0;
        if (performanceRating == 5) bonus = salary * 0.20;
        else if (performanceRating == 4) bonus = salary * 0.10;
        else if (performanceRating == 3) bonus = salary * 0.05;

        cout << fixed << setprecision(2)
             << "Manager Details:\nName: " << name
             << "\nID: " << id
             << "\nBase Salary: $" << salary
             << "\nPerformance Bonus: $" << bonus
             << "\nTotal Earnings: $" << (salary + bonus) << endl;
    }
};

class Developer : public Employee {
    int extraHours;
public:
    Developer(string empName, int empId, double empSalary, int hours)
        : Employee(empName, empId, empSalary), extraHours(hours) {}

    void calculateEarnings() override {
        double overtimeCompensation = extraHours * 20;
        cout << fixed << setprecision(2)
             << "Developer Details:\nName: " << name
```

```
            << "\nID: " << id
            << "\nBase Salary: $" << salary
            << "\nOvertime Compensation: $" << overtimeCompensation
            << "\nTotal Earnings: $" << (salary + overtimeCompensation) << endl;
    }
};

int main() {
    cout << "Choose Employee Type:\n1. Manager\n2. Developer\n";
    int choice;
    cin >> choice;

    string name;
    int id, salary;
    cout << "Enter Employee Name: ";
    cin.ignore();
    getline(cin, name);
    cout << "Enter Employee ID: ";
    cin >> id;
    cout << "Enter Salary: ";
    cin >> salary;

    if (choice == 1) {
        int performanceRating;
        cout << "Enter Performance Rating (1-5): ";
        cin >> performanceRating;

        Manager mgr(name, id, salary, performanceRating);
        mgr.calculateEarnings();
    }
    else if (choice == 2) {
        int extraHours;
        cout << "Enter Extra Hours Worked: ";
        cin >> extraHours;

        Developer dev(name, id, salary, extraHours);
        dev.calculateEarnings();
    }
    else {
        cout << "Invalid employee type selected!" << endl;
    }

    return 0;
}
```

```
C:\Users\Gaurav Kumar\OneD
Choose Employee Type:
1. Manager
2. Developer
1
Enter Employee Name: gaurav
Enter Employee ID: 4663
Enter Salary: 456228
Enter Performance Rating (1-5): 3
Manager Details:
Name: gaurav
ID: 4663
Base Salary: $456228.00
Performance Bonus: $22811.40
Total Earnings: $479039.40

--------------------------------
Process exited after 22.88 seconds with return value 0
Press any key to continue . . .
```

Function Overloading for Calculating Area.

```cpp
#include <iostream>
#include <cmath>
using namespace std;

double area(double radius) {
    return M_PI * radius * radius;
}

double area(double length, double breadth) {
    return length * breadth;
}

double area(double base, double height, bool isTriangle) {
    return 0.5 * base * height;
}

int main() {
    cout << "Choose a shape to calculate area:\n";
    cout << "1. Circle\n2. Rectangle\n3. Triangle\n";
    int choice;
    cin >> choice;

    switch (choice) {
        case 1: {
            double radius;
            cout << "Enter the radius of the circle: ";
            cin >> radius;
            cout << "Area of the circle: " << area(radius) << endl;
            break;
        }
        case 2: {
            double length, breadth;
            cout << "Enter the length and breadth of the rectangle: ";
            cin >> length >> breadth;
            cout << "Area of the rectangle: " << area(length, breadth) << endl;
            break;
        }
        case 3: {
            double base, height;
            cout << "Enter the base and height of the triangle: ";
            cin >> base >> height;
            cout << "Area of the triangle: " << area(base, height, true) << endl;
            break;
        }
        default:
            cout << "Invalid choice! Please select a valid option.\n";
    }
    return 0;
}
```

```
C:\Users\Gaurav Kumar\OneD
Choose a shape to calculate area:
1. Circle
2. Rectangle
3. Triangle
1
Enter the radius of the circle: 5
Area of the circle: 78.5398

--------------------------------
Process exited after 10.5 seconds with return value 0
Press any key to continue . . .
```