Task 01:

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

class toolBooth {
private:
    unsigned int cars;
    double amount;
public:
    toolBooth() : cars(0), amount(0) {}
    void payingCar() {
        cars++;
        amount+=0.50;
    }

    void nopayCar() {
        cars++;
    }

    void display() {
        cout << "Cars: " << cars << endl;
        cout << "Amount: " << amount << endl;
    }
};

int main() {
    toolBooth t1;
    char ch;
    cout << "Press 'P' for a paying car, 'N' for a non-paying car, and ESC to
exit.\n";

    while (true) {
        ch = _getch();  // capture keypress

        if (ch == 27) {  // esc key (ASCII 27) to exit
            t1.display();
            break;
        } else if (ch == 'p' || ch == 'P') {
            t1.payingCar();
            cout << "Paying car counted!\n";
        } else if (ch == 'n' || ch == 'N') {
            t1.nopayCar();
            cout << "Non-paying car counted!\n";
        }
    }
    return 0;
}
```

Task 02:

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

class time {
private:
    int hours;
    int minutes;
    int seconds;
public:
    time() : hours(0), minutes(0), seconds(0) {} // Default
    time(int hrs, int min, int sec) : hours(hrs), minutes(min), seconds(sec)
{} // Parameterized

    void display() const {
        cout << "Time: " << hours << ":" << minutes << ":" << seconds << endl;
    }
    void addTime(time, time);
};

void time::addTime(time t1, time t2) {
    hours = t1.hours + t2.hours;
    minutes = t1.minutes + t2.minutes;
    seconds = t1.seconds + t2.seconds;

    if (seconds >= 60) {
        minutes += seconds / 60;
        seconds %= 60;
    }
    if (minutes >= 60) {
        hours += minutes / 60;
        minutes %= 60;
    }
    hours %= 24; // it's bcz tracking hours in 24-format cause the days aren't
mention in the
}

int main() {
    time t1(12, 20, 40), t2(24, 10, 25), t3(6, 3, 60);
    t1.display();
    t2.display();
    t3.display();
    cout << "=================" << endl;
    t1.addTime(t2, t3);
    t1.display();

    return 0;
}
```

Task 03:

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

class Employee {
private:
    int empNumber;
    float compensation;

public:
    void getData() {
        cout << "Enter Employee Number: ";
        cin >> empNumber;
        cout << "Enter Compensation: ";
        cin >> compensation;
    }

    void display() const {
        cout << "Employee Number: " << empNumber << " | Compensation: $";
        int dollars = compensation;
        int cents = (int)((compensation - dollars) * 100);
        cout << dollars << "." << ((cents) < 10 ? "0" : "") << cents << endl;
    }
};

int main() {
    const int numEmployees = 3;
    Employee employees[numEmployees];

    for (int i = 0; i < numEmployees; i++) {
        cout << "Enter details for Employee " << i + 1 << ":\n";
        employees[i].getData();
    }

    cout << "\nEmployee Details:\n";
    for (int i = 0; i < numEmployees; i++) {
        employees[i].display();
    }

    return 0;
}
```

Task 04:

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

class BankAccount {
private:
    int accnumber;
    float balance;
public:
    BankAccount() : accnumber(0), balance(0) {}
    BankAccount(int acc, float b) : accnumber(acc), balance(b) {}
    void withDrawal(float);
    void deposit(float);
    void transfer(BankAccount&, float);
};
void BankAccount::withDrawal(float amount) {
    if(amount > this->balance) cout << "Insufficient Balance!!" << endl;
    else {
    balance -= amount;
    cout << "Amount: $" << amount << " withdrawal successfully!!";
    }
}
void BankAccount::deposit(float amount) {
    balance += amount;
    cout << "Amount: $" << amount << " deposited successfully to Account " <<
this->accnumber << "!!" << endl;
}
void BankAccount::transfer(BankAccount& b, float a) {
    if(this->balance < a) cout << "Insufficient Balance!!" << endl;
    else {
    this->balance -= a;
    b.balance += a;
    cout << "\nSuccessfully Transfer!!\nAccount Number: " << this->accnumber
<< "\nTransfered Account Number: " << b.accnumber << "\nYour Balance: $" <<
this->balance << "\nReciever Balance: $" << b.balance << endl;
    }
}
int main() {
    cout << "==== Welcome ====" << endl;
    BankAccount b1(1, 200), b2(2, 900), b3(3, 100.33);
    b1.deposit(32.99);
    b2.deposit(32.99);
    b3.deposit(32.99);

    b1.transfer(b2, 30.0);

    return 0;
}
```

Task 05:

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

class Distance {
private:
    int feet, inches;
public:
    Distance() : feet(0), inches(0) {}
    Distance(int f, int i) : feet(f), inches(i) {}

    Distance ADD(const Distance&);
    void display() const;
};

Distance Distance::ADD(const Distance& d) {
    Distance result;
    result.feet = feet + d.feet;
    result.inches = inches + d.inches;

    return result;
}

void Distance::display() const {
    cout << "Total Distance: " << feet << " feet " << inches << " inches." <<
endl;
}

int main() {
    Distance d1(20, 30), d2(10, 80);
    Distance d3 = d1.ADD(d2);
    d3.display();
    return 0;
}
```

Task 06:

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

class Angle {
private:
    int degrees;
    float minutes;
    char direction;

public:
    void getInput() {
        cout << "Enter degrees: ";
        cin >> degrees;
        cout << "Enter minutes: ";
        cin >> minutes;
        cout << "Enter direction (N/S/E/W): ";
        cin >> direction;
    }

    void display() const {
        cout << degrees << "°" << minutes << "' " << direction;
    }
};

class Ship {
private:
    static int counter;
    int serialNumber;
    Angle latitude, longitude;

public:
    Ship() {
        serialNumber = ++counter;
    }

    void setLocation() {
        cout << "Enter latitude for Ship " << serialNumber << ":\n";
        latitude.getInput();
        cout << "Enter longitude for Ship " << serialNumber << ":\n";
        longitude.getInput();
    }

    void display() const {
        cout << "Ship Serial Number: " << serialNumber << "\n";
        cout << "Latitude: "; latitude.display();
        cout << "\nLongitude: "; longitude.display();
        cout << "\n----------------------\n";
```

```
    }
};

int Ship::counter = 0;

int main() {
    const int numShips = 3;
    Ship ships[numShips];

    for (int i = 0; i < numShips; i++) {
        ships[i].setLocation();
    }

    cout << "\nShip Details:\n";
    for (int i = 0; i < numShips; i++) {
        ships[i].display();
    }

    return 0;
}
```