C Prog.

Lec 6

Problem

- Write a program for cashier of a store.
  - The program should keep the data regarding each customer and reciepts.
  - It should be able find total revenue for any time frame.
  - It should be able to list out chronologically sorted list of reciepts for a particular customer.

How to store Data?

- Customer has a
  - Phone No.
  - Name
- Reciept has a
  - date time
  - value
  - customer

How to store a single Customer Data?

```
typedef struct Customer {
    char name[100];
    int phone_no;
} Customer;
```

How to store a single Reciept

Use time_t from time.h

https://en.cppreference.com/w/c/chrono/time_t

```
typedef struct Reciept {
    time_t time;
    float value;
    Customer *customer;
} Reciept;
```

How to store of all the customers and reciepts

```c
typedef struct Database {
    Customer customers[100];
    Reciept reciepts[1000];
} Database;
```

- Assuming max customers < 100 and reciepts < 1000.

Problem

- Suppose we need to add a new customer, where should it go?
- How many customers are there currently?

Solution

```
typedef struct Database {
    Customer customers[100];
    Reciept reciepts[1000];
    int customer_count;
    int reciept_count;
} Database;
```

- `customer_count` stores the current number of customers. New customers are added to the `customers[customer_count]` and `customer_count` is incremented.

Wrong

```
Customer* add_customer(char *name,
                       int phone_no,
                       Database *db) {
    Customer c = db->customers[db->customer_count];
    c.phone_no = phone_no;
    c.name = name;
    db->customer_count++;
    return c;
}
```

Right approach

```
Customer* add_customer(char *name,
                       int phone_no,
                       Database *db) {
    Customer *c =
      &(db->customers[db->customer_count]);
    c->phone_no = phone_no;
    strcpy(c->name, name);
    db->customer_count++;
    return c;
}
```

Add Payment Mode to Reciept

Payment Mode as `int`

```
typedef struct Reciept {
    time_t time;
    float value;
    Customer *customer;
    int payment_mode;
} Reciept;
```

Problem

- We have to remember that
  - `0` is for cash,
  - `1` is for card,
  - `2` is for upi
- Code will contain these magic numbers, which might not be obvios to another person.

Enum

```
typedef enum PayMode {
    Cash = 0,
    Card = 1,
    UPI = 2
} PayMode;

typedef struct Reciept {
    time_t time;
    float value;
    Customer *customer;
    PayMode mode;
} Reciept;
```