Objective:

- It will help in understanding the use of struct and passing it by reference/value and to deal with array of struct objects.

Challenge – 1: *Time Calculator* (5)

In this challenge you will be writing a function which calculate difference between two times, actually, the difference between system time and user provided time.
The function that you are required to implement is named as `getTimeDifference`, which receives a `struct time` object and calculates its time difference with the system time. The struct Time is also defined below in which time is stored in 24-hour format:

```
struct Time
{
    int hours;
    int minutes;
    int seconds;
};
```

Function Prototype: `Time getTimeDifference ( Time );`

Sample Run:

1	`Time t = { 10, 15, 50};`
	`//assume while execution of this sample run, the system time is 22:50:12`
	`Time diffTime = getTimeDifference(t);`
	`cout << diffTime.hours << " : " << diffTime.minutes << " : " << diffTime.seconds;`
	`//Prints 12 : 34 : 22`

2	`Time t = { 10, 15, 50};`
	`//assume while execution of this sample run, the system time is 10:5:2`
	`Time diffTime = getTimeDifference(t);`
	`cout << diffTime.hours << " : " << diffTime.minutes << " : " << diffTime.seconds;`
	`//Prints 0 : 10 : 48`

But you may only implement this function if you know the way to find system time (the current time on your computer).
When you will execute the following code, It will store the system time in your Time object named below currentTime. We don't need to know much about the code given below just paste the code snippet in your function to find the system time. In near future, you will know each and every bit of it hopefully.

Code to get System Time	Short Explanation
`Time currentime;`	Time object in which system time will be stored.
`time_t t = time(0);`	The C library function ***time_t time(time_t *seconds)*** returns the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds.
`tm curTime = * localtime(&t);`	localtime function converts given time since epoch as std::time_t value into calendar time, expressed in local time. tm structure holding a calendar date and time broken down into its components.

`currentDate.day = curTime.tm_hour;`
`currentDate.month = curTime.tm_min;`
`currentDate.year = curTime.tm_second;`
	Storing the system time into your Time object

In order for the above code to work, you need to include ctime library.

```
#include<ctime>
```

Challenge – 2: *Token Machine & Service Counters* (21.5)

Our main task for today's lab in which we are going to design a system for generating tokens that you usually see when you visit some where you get the token and wait in a queue to get the required work done at some particular counter.
We shall be using a bunch of structs for achieving our task.

```
struct Date
{
    int day;
    int month;
    int year;
};
```

```
struct Time
{
    int hours;
    int minutes;
    int seconds;
};
```

```
struct Token
{
    int tokenNumber;
    Date tokenDate;
    Time tokenTime;
};
```

```
struct ServiceCounter
{
    int counterNumber;
    Token list[100];
    int tokensCount;
};
```

Purpose of structs given above is as follows:

Date	Use to store date information.
Time	Use to store time information in 24-hour format.
Token	It represents a token we that use in our daily life. Its attributes consist of the token number and date and time on which this token is generated. Token number is the one attribute which will decide who comes first in the office and will be serviced first.
ServiceCounter	It represents the counter at which a person with token will reach to get its service done. ServiceCounter maintains a queue of tokens and will service tokens on the basis of first come first serve. There can be multiple counters at a place. A ServiceCounter object will not have more than 100 token at one time in his queue that's the reason why an array of 100 tokens is created in it. The member tokensCount represents the token currently waiting to be serviced.

Following Operations to be defined for different stucts:

1. `void printDate ( Date );`
   It prints the date information on console.
2. `void printTime ( Time );`
   It prints the time information on console.
3. `Token generateToken ( );`
   This function is responsible for generating token such that every time when this function is called it creates an object of Token and initialize it with a unique token umber which is never generated before and initialize with current system date and time. See sample run # 1 to understand further.
   You already know in challenge # 1 that how to get system time. I am dumping code below that will extract the system date as well.

```
Date currentDate;
time_t t = time(0);
tm curTime = * localtime(&t);
currentDate.day = curTime.tm_mday;
currentDate.month = curTime.tm_mon + 1;
currentDate.year = curTime.tm_year + 1900;
```

4. `void printToken ( Token );`
   It prints the token information on console. See sample run # 1.
5. `void initializeServiceCounter ( ServiceCounter & );`
   It initializes the received ServiceCounter object by assigning it a unique service counter number. It also initializes the tokenCount to 0. See sample run # 2.
6. `void assignToken ( ServiceCounter & , Token );`
   It assigns a token object to be serviced at a given ServiceCounter. See sample run # 2.
7. `void serviceToken ( ServiceCounter & );`
   It serive the token with least number (First Come First Serve) in the received ServiceCounter object. See sample run # 2.

8. `void printTokenAtServiceCounter ( ServiceCounter );`
   It prints all the tokens that are in queue waiting to get service at received counter. See sample run # 2.

SR #	Sample Runs	
	Sample Code	**Console Output**

1	`Token x = generateToken();`
	`Token y = generateToken();`
	`printToken(x);`
	`printToken(y);`
		`Token # : 1`
		`Time: 26:1:2024`
		`Date: 1:0:7 AM`
		`Token # : 2`
		`Time: 26:1:2024`
		`Date: 1:0:7 AM`

2	`ServiceCounter a, b;`
	`initializeServiceCounter(a);`
	`initializeServiceCounter(b);`
	
	`assignToken(a, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(a, generateToken());`
	`assignToken(a, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(b, generateToken());`
	`assignToken(a, generateToken());`	

	`printTokensAtServiceCounter(a);`	`*** Token Queue Waiting For Service @ Counter # 1`
		`Token # : 1`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 3`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 4`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 10`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`

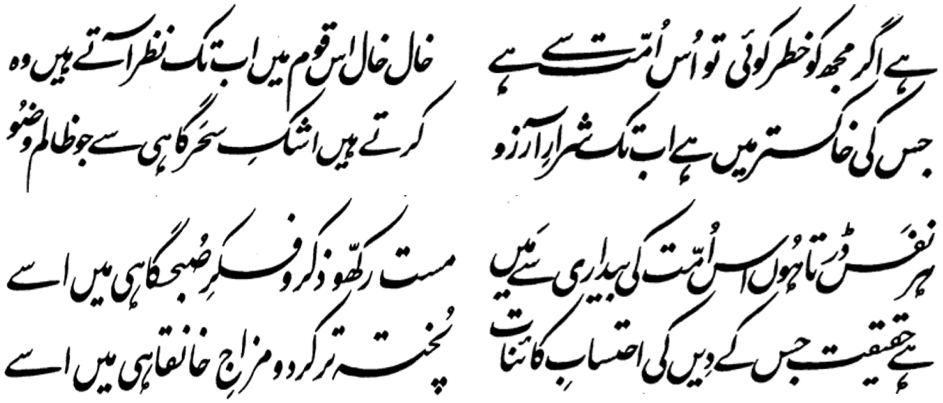
	`serviceToken(a);`	

	`printTokensAtServiceCounter(a);`	`*** Token Queue Waiting For Service @ Counter # 1`
		`Token # : 3`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 4`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 10`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`

	`serviceToken(a);` `serviceToken(a);` `serviceToken(a);`	
	`printTokensAtServiceCounter(a);`	`*** Token Queue Waiting For Service @ Counter # 1` `--> Queue is Empty <---`
	`serviceToken(b);`	

	`printTokensAtServiceCounter(b);`	`*** Token Queue Waiting For Service @ Counter # 2`
		`Token # : 5`
		`Time: 26:1:2024`
		`Date: 12:43:49 AM`
		
		`Token # : 6`

		Time: 26:1:2024 Date: 12:43:49 AM Token # : 7 Time: 26:1:2024 Date: 12:43:49 AM Token # : 8 Time: 26:1:2024 Date: 12:43:49 AM Token # : 9 Time: 26:1:2024 Date: 12:43:49 AM



ابلیس کی مجلس شوریٰ

ارمغان حجاز

علامہ محمد اقبال