



# Assignment 3

<b>Course</b>	<b>Programming Fundamentals</b>
<b>Instructor</b>	<b>Ms. Hina Iqbal</b>
<b>Section</b>	<b>BCS-1E, BCS-1F</b>

## ***Important Guidelines:***

1. If you use **AI tools** (ChatGPT, Copilot, etc.) or **copy code from other students/online sources**, your assignment will be given **zero marks** without exception.

**Prophet Muhammad (SAW) said: “Truthfulness leads to righteousness, and righteousness leads to Paradise” - Sahih Al Bukhari 6094**

2. Submit your assignment **on time**. Late submissions will incur a 25%-mark deduction if submitted within 6 hours, a 50%-mark deduction if submitted within 24 hours, and submissions beyond 24 hours will result in a score of zero.
3. No marks will be awarded to individuals who do not submit the file on Google Classroom (GCR).

4. Only submit your cpp files on google classroom. **Do not zip your files.** Each cpp file should be renamed according to the format: **I25xxxx\_q1.cpp**. **Failure to follow this format will result in a score of zero.**
  5. In case of confusion please feel free to contact on email [I227910@lhr.nu.edu.pk](mailto:I227910@lhr.nu.edu.pk) or [hina.iqbal@nu.edu.pk](mailto:hina.iqbal@nu.edu.pk)
- 

## Questions

---

### Q1. Function Overloading – Temperature Converter

Create an overloaded function `convert()` that:

- Converts **Celsius to Fahrenheit**
- Converts **Fahrenheit to Celsius**

**Requirements:**

- Two overloaded versions:
  - `float convert(float celsius)` → Fahrenheit
  - `float convert(float fahrenheit, bool isFahrenheit)` → Celsius
- Display both conversions for given values.

**Example Input:**

```
Celsius: 37
Fahrenheit: 100
```

**Output:**

```
37°C = 98.6°F
100°F = 37.78°C
```

---

### Q2. Employee Bonus Calculation

Develop a modular program that uses **arrays and functions** to compute employee bonuses.

**Rules:**

- Input number of employees (max 10)
- Input their salaries
- Bonus rules:
  - Salary < 50,000 → 10%
  - 50,000–100,000 → 7%
  - 100,000 → 5%
- Display a formatted table with each employee's salary and bonus.

**Example Input:**

```
3
45000
75000
120000
```

**Output:**

Salary	Bonus
45000	4500
75000	5250
120000	6000

**Edge Case:** Negative salary or zero employees → display “Invalid data.”

---

### **Q3. Library Late Fee System**

Write a function-based program to compute total late fees for **N** books returned late.

**Rules:**

- Fee = Rs. 10/day for first 5 days, Rs. 20/day afterward.
- Input: Number of books, and delay (in days) for each.
- Output total fine.

**Sample Input:**

```
3  
2 7 10
```

**Output:**

```
Total Fine = Rs. 310
```

**Edge Case:** No late books (0 days delay) → Rs. 0 fine.

---

## Q4. Student Grades Analyzer

Create a program that:

- Takes marks of **N** students (max 100).
- Finds **highest**, **lowest**, **average**, and **standard deviation**.
- Uses **separate functions** for each operation.

**Sample Input:**

```
5  
60 70 80 90 100
```

**Output:**

```
Highest: 100  
Lowest: 60  
Average: 80  
Standard Deviation: 14.14
```

---

## Q5. Array Searching and Sorting System

Implement a program that:

1. Inputs an array of integers (max 50 elements).
2. Allows the user to choose an operation:
  - o (1) Linear Search
  - o (2) Binary Search
  - o (3) Bubble Sort

3. Display the result or sorted array.

**Requirements:**

- Use **separate functions** for each operation.
- Implement **binary search** only on sorted arrays.

**Example Input:**

```
6  
4 2 9 1 5 6  
Choose operation: 3
```

**Output:**

```
Sorted Array: 1 2 4 5 6 9
```

## Q6. String Frequency Counter

Write a program that counts the frequency of each vowel in a string.

**Example Input:**

```
Enter text: Programming Fundamentals
```

**Output:**

```
A: 2  
E: 0  
I: 1  
O: 1  
U: 1
```

## Q7. Password Strength Checker

Design a C++ program to evaluate the strength of a password using **character arrays**.

**Rules:**

- Password length must be  $\geq 8$ .
- Must include at least:
  - One uppercase letter
  - One lowercase letter
  - One digit
  - One special character (!,@,#,\$,%,&)
- Display: “Weak”, “Moderate”, or “Strong”

**Example Input:**

```
Password: Abc@1234
```

**Output:**

```
Password Strength: Strong
```

**Edge Case:** Shorter than 8 → “Invalid Password.”

---

## Q8. Flight Scheduling System

You are developing a small **flight scheduler** for a travel agency.

**Requirements:**

- Input number of flights (max 20).
- For each flight, input **departure time** (24 hours format -> hh:mm), **arrival time**, and **destination city**.
- Compute the **duration** of each flight (in hours and minutes).
- Sort flights by **departure time** using any sorting algorithm.
- Display all flights with their departure time, duration and destination.

**Example Input:**

```
3
08:00 10:00 Karachi
11:30 15:00 Lahore
06:45 09:30 Islamabad
```

**Output:**

```
Sorted by Departure:
```

```
06:45 Islamabad (Duration: 2h 45m)
08:00 Karachi    (Duration: 3h 30m)
11:30 Lahore     (Duration: 1h 15m)
```

**Hint:** Use string manipulation and integer conversion for time

---

## Q9. Maximum Balanced Subarray

A *balanced subarray* is defined as a contiguous portion of an integer array where the number of even and odd elements is equal.

Write a program to find the **length of the longest balanced subarray**.

### Requirements:

- Input: Size of array **N** and **N** integers.
- Output: Length of the longest balanced subarray.
- Must use **functions** for subarray detection and validation.

### Example Input:

```
8
2 3 4 1 6 5 8 10
```

### Output:

```
Longest Balanced Subarray Length: 6
```

### Explanation:

Balanced subarray = [3,4,1,6,5,8] → 3 even, 3 odd.

### Edge Cases:

- No balanced subarray → output 0.
- All even or all odd → output 0.

---

Good Luck