AL-KAWTHAR UNIVERSITY

Department of Computer Science

CS 121 L – Programming Fundamentals (PF)

Lab # 03

Objective:

To introduce students to loops and relational operators (<, <=, >, >=, ==, !=). Students will learn to implement loops (for, while, do-while) and understand their appropriate use cases.

Name of Student	**Muhammad Hashir Rafique**
Student ID	**BSCS24101023**
Date of Lab Conducted	
Marks Obtained	
Remarks	
Signature	

LAB 3 - ACTIVITY 1

Generating a Table of 2 Without Using a Loop

Objective:

- Demonstrate the repetitive nature of loops by manually generating a table of 2.

Activities

1. Manually compute the table of 2 from 1 to 5:

- printf("2 × 1 = 2\n");
- printf("2 × 2 = 4\n");
- printf("2 × 3 = 6\n");
- printf("2 × 4 = 8\n");
- printf("2 × 5 = 10\n");

2. Discuss the inefficiency of this approach for larger ranges.

```
#include <stdio.h>

int main() {
    printf("2 × 1 = 2\n");
    printf("2 × 2 = 4\n");
    printf("2 × 3 = 6\n");
    printf("2 × 4 = 8\n");
    printf("2 × 5 = 10\n");
    return 0;
}
```

Why This is Inefficient

Writing each line manually works only for small numbers. But imagine printing the table of 2 from 1 to 100. You'd need to write `printf("2 × n = result\n");` 100 times. That wastes time, space, and increases chances of mistakes.

LAB 3 - ACTIVITY 2

Relational Operators with Example Code

Objective:

- Explain relational operators and their use in loop conditions.

Activities

Define relational operators with examples:

- < (Less than): if (a < b)
- <= (Less than or equal): if (a <= b)
- > (Greater than): if (a > b)
- >= (Greater than or equal): if (a >= b)
- == (Equal to): if (a == b)
- != (Not equal to): if (a != b)

Provide code snippets demonstrating each operator.

Example Code:

```
#include <stdio.h>
int main() {
      int a = 5, b = 10;
      if (a < b) {
            printf("%d is less than %d\n", a, b);
      }
      return 0;
}
```

Code Examples (Every operator)

```
#include <stdio.h>

int main() {
    int a = 5, b = 10;

    if (a < b) {
        printf("%d is less than %d\n", a, b);
    }

    if (a <= b) {
        printf("%d is less than or equal to %d\n", a, b);
    }

    if (a > b) {
        printf("%d is greater than %d\n", a, b);
    }

    if (a >= b) {
        printf("%d is greater than or equal to %d\n", a, b);
    }

    if (a == b) {
        printf("%d is equal to %d\n", a, b);
    }

    if (a != b) {
        printf("%d is not equal to %d\n", a, b);
    }

    return 0;
}
```

LAB 3 - ACTIVITY 3

Generating a Table of 2 Using a For Loop

Objective:

- Implement a loop to automate repetitive tasks.

Activities:

1. Write pseudocode for generating a table of 2 using a for loop:

```
START
FOR i = 1 TO 5
    result = 2 * i
    PRINT "2 ×", i, "=", result
END FOR
END
```

2. Draw a flowchart for the above logic.

Pseudocode

```
START
FOR i = 1 TO 5
    result = 2 * i
    PRINT "2 ×", i, "=", result
END FOR
END
```

FLOWCHART OF THE PREVIOUS CODE

LAB 3 - ACTIVITY 4

While and Do-While Loop Demonstration

Objective:

- Compare while and do-while loops and their use cases.

Activities:

1. Explain when to use each loop:

- For loop: When the iteration count is known.
- While loop: When the end condition is known but iteration count is not.
- Do-while loop: When the loop must execute at least once.

2. Provide pseudocode examples for each loop type.

Example:

```
#include <stdio.h>

int main() {
    int n = 5, i = 1;

    while (i <= n) {
        printf("2 × %d = %d\n", i, 2 * i);
        i++;
    }

    return 0;
}
```

Example:

```
#include <stdio.h>

int main() {
    int n = 5, i = 1;

    do {
        printf("2 × %d = %d\n", i, 2 * i);
        i++;
    } while (i <= n);

    return 0;
}
```

---

1. Usage of Loops

Feature	`for` loop	`while` loop	`do-while` loop
Condition check	Before first run	Before first run	After first run
Runs at least once?	Only if condition true	Only if condition true	Yes, always once
Use case	Fixed repetitions	Unknown repetitions	Menu systems, validations

2. Pseudocode Examples for each loop type.

for loop

```
START
FOR i = 1 TO 5
    PRINT 2 × i = result
END FOR
END
```

while loop

```
START
Set i = 1
WHILE i <= 5
    PRINT 2 × i = result
    i = i + 1
END WHILE
END
```

do-while loop

```
START
Set i = 1
DO
    PRINT 2 × i = result
    i = i + 1
WHILE i <= 5
END
```

LAB 3 - ASSIGNMENT

Write C programs to:

1. Generate tables of user-input numbers using all three loops.
2. Check if a number is positive/negative using relational operators.
3. Submit pseudocode, flowcharts, and hand-drawn diagrams.

---

(1)

for loop

```
#include <stdio.h>

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    for (int i = 1; i <= 10; i++) {
        printf("%d × %d = %d\n", num, i, num * i);
    }

    return 0;
}
```

FLOWCHART OF THE PREVIOUS CODE

while loop

```
#include <stdio.h>

int main() {
    int num, i = 1;
    printf("Enter a number: ");
    scanf("%d", &num);

    while (i <= 10) {
        printf("%d × %d = %d\n", num, i, num * i);
        i++;
    }

    return 0;
}
```

FLOWCHART OF THE PREVIOUS CODE

do-while loop

```
#include <stdio.h>

int main() {
    int num, i = 1;
    printf("Enter a number: ");
    scanf("%d", &num);

    do {
        printf("%d × %d = %d\n", num, i, num * i);
        i++;
    } while (i <= 10);

    return 0;
}
```

FLOWCHART OF THE PREVIOUS CODE

Negative / Positive Number Checker

```
#include <stdio.h>

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (num > 0) {
        printf("The number is positive.\n");
    } else if (num < 0) {
        printf("The number is negative.\n");
    } else {
        printf("The number is zero.\n");
    }

    return 0;
}
```

FLOWCHART OF THE PREVIOUS CODE

Pseudocode(s):

for loop

```
START
INPUT number
FOR i = 1 to 10
    PRINT number × i = result
END FOR
END
```

while loop

```
START
INPUT number
SET i = 1
WHILE i <= 10
    PRINT number × i = result
    i = i + 1
END WHILE
END
```

do-while loop

```
START
INPUT number
SET i = 1
DO
    PRINT number × i = result
    i = i + 1
WHILE i <= 10
END
```

Negative / Positive Number Checker

```
START
INPUT number
IF number > 0
    PRINT "Positive"
ELSE IF number < 0
    PRINT "Negative"
ELSE
    PRINT "Zero"
END IF
END
```