Programming Lab Report

Vivid Padungkiatsakul 6601023620026[1]

[1]Robotic Engineering and Automation System, Faculty of Engineering, King Mongkut's University of Technology North Bangkok

[010243107] C Programming 2023

Contents

[Blank page]
Go on to the next page for Mid-term Experiment

Lab Week 1

Flowchart, printf and scanf

Topic

- Introduction
- Problem analysis
- Flowchart
- Compilation and running a program
- printf/scanf

Experiment I

---

1 Problem description

- Write a program to show message
- ”Hello xxxxxxxxx on screen, where xxxxxxxxx is your name.

2 Program design

- use printf; function to output text in terminal.

3 Program text

```c
#include <stdio.h>

int main() {
  printf("Hello Vivid");
  return 0;
}
```

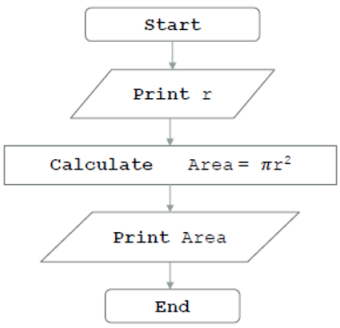
Experiment II

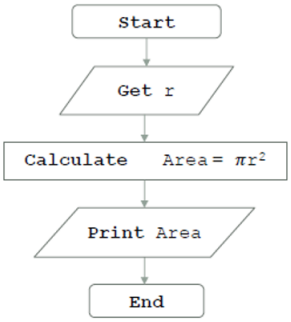
---

1 Problem description

- According to flowcharts below, write program for each of them.

2 Problem Flowchart





3 Program design

- First, include math.h for use Pi.
- For Left flowchart r variable is fix. so we can set r in any number.
- For Right flowchart r variable is from user input. so, we use scanf; to get user input from terminal.

4 Program Text

```c
#include <stdio.h>
#include <math.h>
#define M_PI 3.14159265358979323846

int main(){
    float r = 20;
    float area;
    area = M_PI*r*r;
    printf("%.6f\n",r);
    printf("%.6f",area);
    return 0;
}
```

Ex2-1.c

```c
#include <stdio.h>
#include <math.h>
#define M_PI 3.14159265358979323846

float r,area;

int main(){
    printf("INSERT R: ");
    scanf("%f",&r);
    area = M_PI * r * r;
    printf("%f\n",r);
    printf("%f",area);
}
```

Ex2-2.c

Experiment III

---

1 Problem description

- Answer following questions..

Q 3.1: Run lab1-1.c and observe result. What is the effect of \n and \t?

Q 3.2: Try adding \n after double quote in lab1-1.c as shown below.

```
printf("He lives in Bangkok \n);
```

Save, compile, run and observe the result. What is error message? How to correct it?

2 Problem program text

```
#include <stdio.h>
int main (void)
{
   printf("Somchai\n is very handsome.");
   printf("He is 20 years \nold.");
   printf("He lives\tin Bangkok .");
}
```

3 Program design

- \n use to enter new line in terminal.
- \t use to add tab space in terminal.

4 Answer

Ans 3.1: \n use to enter the new line and \t use to add space like [tab]

Ans 3.2: In this case the error will get is:

```
main.c:6:33: error: stray '\' in program
    6 |     printf("He lives in Bangkok" \n);
      |                                  ^
main.c:6:32: error: expected ')' before 'n'
    6 |     printf("He lives in Bangkok" \n);
      |           ~                     ^ ~
      |                                 )
```

So, to fix this Error. We have to move \n before double quote (")

Experiment IV

---

1 Problem description

- Answer following questions..

Q4.1: What values are shown correctly as an integer, a floating point, and a character?

Q4.2: Compare the result of %d and %5d. What is the effect of 'n' in %nd?

Q4.3: Compare the result of %f and %.4f. What is the effect of '.4' in %.4f?

Q4.4: Compare the result of %5.2f and %10.3f. What is the effect of 'n' and 'm' in %n.mf?

Q4.5: Compare the result of %17.4f and %-17.4f. What is the different between having and having no negative sign?

2 Problem program text

```c
#include <stdio.h>
int main(void)
{

    printf("--------------Question 1--------------------------------------\n");
    printf("Test output values\n");
    printf("an integer  :%d %d %d %d\n", 10,   -8, 0.5, 'a');
    printf("a float     :%f %f %f %f\n", 9.5,1.54,   7, 'd');
    printf("a character :%c %c %c %c\n", 'b', 'c',  70, 2.5);
    printf("--------------Question 2--------------------------------------\n");
    printf("%d\n", -71);
    printf("%5d\n", -71);
    printf("%d\n", 8800);
    printf("%5d\n", 8800);
    printf("%d\n", 123);
    printf("%5d\n", 123);
    printf("--------------Question 3--------------------------------------\n");
    printf("%f\n", 1.23456);
    printf("%.4f\n", 1.23456);
    printf("--------------Question 4 --------------------------------\n");
    printf("%5.2f\n", 2.58321);
    printf("%5.2f\n", 2.5);

    printf("%10.3f\n", -9.6357);
    printf("%10.3f\n", -9.6);
    printf("--------------Question 5--------------------------------\n");
    printf("%17.4f\n", -8.05715557);
    printf("%-17.4f\n", -8.05715557);

    return 0;
}
```

3 Answer

Ans4.1: Except the Character because there are Float and ASCII can't convert Float to Character. So the result will be:

Integer: [10 -8 0 97], Floating-Point: [9.500000 1.540000 0.000000 0.000000], and Character: [b c F]

If we want to show all output in Character, we have to change 2.5 to other value that doesn't be Float

Ans4.2: The effect of n (5) is the output of int in n width.

Ans4.3: The effect of .4 in %.4f is use only 4 digits of float and round the last digit.

Ans4.4: The effect between n (10) and m (.3) is the n is output a floating-point of n width. The m is how many digits of float.

Ans4.5: The effect on - and no - is what place to place the space. In (-) is place on the left (after the float).

End of Lab Experiment Week 1

Lab Week 2

Statement, Variables and Expressions

Topic

- Structure of C program
- Statement
- Comment
- Variables
- Expressions
- Operators

Experiment I

---

1 Problem description

- Write a program that stores data and also display them on the screen as follow:

2 Problem terminal text

```
Height of Somchai: 126 cm
Grade of Somjai: B
Age of Winai: 20
Weight of Anna: 50.3 kg
Gravity constant: 9.81 m/s^2
Distance from Bangkok to Nonthaburi: 53 km
```

3 Program design

- int use to make variable whole number that doesn't have decimal.
- float use to make variable whole number that have decimal.
- char use to make variable whole the character.
- We can set multiple variables in one line if that variables are same data type.

4 Program text

```c
#include <stdio.h>

int hight= 126, age = 20,distance = 53;
char grade = 'B';
float weight = 50.3, gravity = 9.81;

int main(){
    printf("Height of Somchai: %d cm\n", hight);
    printf("Grade of Somjai: %c\n", grade);
    printf("Age of Winai: %d\n", age);
    printf("Weight of Anna: %.1f kg\n", weight);
    printf("Gravity constant: %.2f m/s^2\n", gravity);
    printf("Distance from Bangkok to Nonthaburi: %d km\n", distance);
    return 0;
}
```

Experiment II

---

1 Problem description

- Write a program that display result of

$$x * y - 20\%z$$

  when $x = 2,\ y = 7,\ z = 4$.

- Explain how to solve this by hand.

2 Program design

- The % is get the number that remain from division.

3 Equation solve

- so,first insert x y z in equation:

$$2 * 7 - (20\%4)$$

  Then, solve 20%5:

$$20\%4 = 0$$

  Finally, take the result from 20%5 insert in the equation:

$$2 * 7 - 0 = 14$$

  ∴The answer of these equation is 14

4 Program text

```c
#include <stdio.h>

int x = 2, y = 7, z = 4, result;

int main(){
    result = x * y - 20 % z;
    printf("Answer is %d", result);
    return 0;
}
```

5 Terminal output

```
Answer is  14
```

Experiment III

---

1 Problem description

- Write a program that displays result of equation below.

$$result = \frac{4.2a + 2.8b}{\frac{5b}{a} - 7c}$$

- Also, verify your program.

2 Program design

- result is decimal. so, We set result in float variable
- For program design. We will get $a,\ b,\ c$ from user input.
- For verify program. We will set $a,\ b,\ c = 1$, and compare to answer from hand solve.

3 Equation solve

- so,first insert $a\ b\ c = 1$ in equation:

$$result = \frac{4.2(1) + 2.8(1)}{\frac{5(1)}{(1)} - 7(1)}$$

Then, solve equation:

$$result = \frac{4.2 + 2.8}{5 - 7}$$
$$result = \frac{7}{-2}$$
$$result = -3.5$$

4 Program text

```c
#include <stdio.h>

float a,b,c;
float result;
int test;

int main(){
    float a=1,b=1,c=1; //use for testing
    // printf("Enter value of a b and c: ");
    // scanf("%f %f %f", &a, &b, &c); // For user input
    result = ((4.2*a)+(2.8*b))/(((5*b)/a)-(7*c));
    printf("%f\n",result);
    test = (result == -3.5);
```

```
    //return output in True or False
    switch (test) {
        case 1:
            printf("True");
            break;
        case 0:
            printf("False");
            break;
    }
    return 0;
}
```

5 Terminal output

- User input value

```
Enter value of a b and c: 1 1 1
-3.500000
```

- Testing result:

```
-3.500000
True
```

End of Lab Experiment Week 2

Lab Week 3

Condition, If-else and Switch statement

Topic

- Arithmetic and Logical comparision
- Condition and operation
- If-else statement
- Switch statement

Experiment I

---

1 Problem description

- Write a program that gets value of x and displays the value of function f.

$$Given\ f(x) = \begin{cases} x^2 + 5, & when\ x < -2 \\ \frac{2.5}{x^3}, & when\ x \geq -2 \end{cases}$$

Also, verify your program with hand calculation.

2 Program design

- The function have 2 case when x in different value.
- Use math.h to use pow for power equation.
- For coding, we have to use if-else to split in 2 case. One is $x < -2$ and other is $x \geq 2$
- For verify program. We will set $x = 3$ and $x = -3$, and compare to answer from hand solve.

3 Equation solve

- Case [1] $x = 3$:
  insert $x = 3$ in $f(x)$

$$f(3) = \frac{2.5}{(3)^3}$$

Then, solve equation:

$$f(3) = \frac{2.5}{27}$$
$$f(3) \approx 0.09259259259$$

- Case [2] $x = -3$:
  insert $x = -3$ in $f(x)$

$$f(-3) = (-3)^2 + 5$$

Then, solve equation:

$$f(3) = 9 + 5$$
$$f(3) \approx 14$$

Part 3: Program text is on the next page.

4 Program text

```c
#include <stdio.h>
#include<math.h>

int main() {
    float x;
    printf("Insert value x: ");
    scanf("%f", &x);
    if (x < -2)
    {
        printf("%f",pow(x,2)+5);
    } else if (x >= -2)
    {
        printf("%f",2.5/pow(x,3));
    }
    return 0;
}
```

5 Terminal output

- Case 1

```
Insert value x: 3
0.092593
```

- Case 2

```
Insert value x: -3
14.000000
```

Experiment II

---

1 Problem description

- Write a program that gets age of visitor and displays zoo entrance fee. The fee is free for person who is 65 years or older and is 100 Baht for person who is younger than 65. The screen should show message as shown below.

  First run:

```
Enter age: 65
Entrance fee: 0 Baht
```

  Second run:

```
Enter age: 30
Entrance fee: 100 Baht
```

2 Program design

- There are 2 event. when age are ≥ 65 and < 65.
- So, if we set in function, it will be $f(x_{age}) = \begin{cases} Free, & when\ x_{age} \geq 65 \\ 100, & when\ x_{age} < 65 \end{cases}$
- We use if-else for split case.

3 Program text

```c
#include <stdio.h>
#include<math.h>

int main() {
    int x;
    printf("Enter age: ");
    scanf("%d", &x);
    if (x >= 65)
    {
        printf("Entrance fee: 0 Baht");
    } else {
        printf("Entrance fee: 100 Baht");
    }
    return 0;
}
```

Experiment III

---

1 Problem description

- By using **if-else if condition**, write a program that gets score from user and displays grade. If score is greater than 100 or less than 0, grade is invalid.

Score	Grade
>100	Invalid
85-100	A
70-84	B
55-69	C
0-54	F
<0	Invalid

- Also, verify your program.

2 Program design

- The function will be:

$$f(x_{score}) = \begin{cases} Invalid, & when\ x_{score} > 100\ or\ x_{score} < 0 \\ A, & when\ 85 \leq x_{score} \leq 100 \\ B, & when\ 70 \leq x_{score} \leq 84 \\ C, & when\ 55 \leq x_{score} \leq 69 \\ F, & when\ x_{score} < 54 \end{cases}$$

- In this case we have to use if condition to split case.
- For coding. We will get score from user input.
- For testing and evaluate, We will make 2 function: ScoreCal function and Test function
- We include stdlib.h for use rand() to random number.
- For random in range, We have to % The range of number and + the min.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#include<math.h>

int main() {
    int x;
    printf("Score: ");
    scanf("%d", &x);
    if (x > 100 || x < 0) {
        printf("Invalid");
    } else if (x >= 85) {
        printf("A");
    } else if (x >= 70) {
        printf("B");
    } else if (x >= 55) {
        printf("C");
    } else {
        printf("F");
    }
    return 0;
}
```

Main Program

```c
#include <stdio.h>
#include <stdlib.h>

//score cal function
const char* ScoreCal(int score){
    if (score > 100 || score < 0) {
        return "Invalid";
    } else if (score >= 85) {
        return "A";
    } else if (score >= 70) {
        return "B";
    } else if (score >= 55) {
        return "C";
    } else {
        return "F";
    }
}

//test function
void Test(){
    printf("%d",ScoreCal(101)=="Invalid")
        ;
    printf("%d",ScoreCal(rand() % 16 +
        85)=="A");
    printf("%d",ScoreCal(rand() % 15 +
        70)=="B");
    printf("%d",ScoreCal(rand() % 10 +
        55)=="C");
    printf("%d",ScoreCal(rand() % 55)=="F
        ");
    printf("%d",ScoreCal(-1)=="Invalid");
}

int main() {
    Test();
}
```

Test Program

Experiment IV

---

1 Problem description

- By using **switch statement**, write a program that gets weight of an orange and shows orange size.

Weight (g)	Size
401-500	Extra large
301-400	Large
201-300	Medium
101-200	Small
0-100	Tiny

- Also, verify your program.

2 Program design

- The function will be:

$$f(x_{weght}) = \begin{cases} Extra\ large, & x_{weight} \in \{401 \leq x_{weight} \leq 500\} \\ Large, & x_{weight} \in \{301 \leq x_{weight} \leq 400\} \\ Medium, & x_{weight} \in \{201 \leq x_{weight} \leq 300\} \\ Small, & x_{weight} \in \{101 \leq x_{weight} \leq 200\} \\ Tiny, & x_{weight} \in \{0 \leq x_{weight} \leq 100\} \end{cases}$$

- In this case we have to use switch condition to split case.
- For coding. We will get weight from user input.
- For testing and evaluate, We will make 2 function: Cal function and Test function
- We include stdlib.h for use rand() to random number.
- For random in range, We have to % The range of number and + the min.
- In this Experiment, The range length are the same. So, We can make fix function.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#include <math.h>

int main() {
    int x;
    printf("Weight: ");
    scanf("%d", &x);
    switch (x){
    case 0 ... 100:
        printf("Tiny");
        break;
    case 101 ... 200:
        printf("Small");
        break;
    case 201 ... 300:
        printf("Medium");
        break;
    case 301 ... 400:
        printf("Large");
        break;
    case 401 ... 500:
        printf("Extra large");
        break;
    }
    return 0;
}
```

Main Program

```c
#include <stdio.h>
#include <stdlib.h>

const char* Cal(int x){
    switch (x){
    case 0 ... 100:
        return "Tiny";
        break;
    case 101 ... 200:
        return "Small";
        break;
    case 201 ... 300:
        return "Medium";
        break;
    case 301 ... 400:
        return "Large";
        break;
    case 401 ... 500:
        return "Extra large";
        break;
    }
    return 0;
}

int random(int min, int max){
    int result =  rand() % (max - min +
        1) + min;
    return result;
}

void Test(){
    printf("%d",Cal(random(0,100))=="Tiny
        ");
    printf("%d",Cal(random(101,200))=="
        Small");
    printf("%d",Cal(random(201,300))=="
        Medium");
    printf("%d",Cal(random(301,400))=="
        Large");
    printf("%d",Cal(random(401,500))=="
        Extra large");
}

int main() {
    Test();
}
```

Test Program

End of Lab Experiment Week 3

Lab Week 4

Loop operation and Condition Statement

Topic

- while
- do..while
- For
- Nested loop
- Break/Continue

Experiment I

---

1 Problem description

- By using **while loop**, write a program that gets 5 value from user and shows the sum value.
- Also, verify your program.

2 Program design

- So we set 3 variables. count[For counting loop], user[for user input value] and sum[for sum number]
- For program, we will get user input
- For verify and testing. we will use x[5] set. In x will have {1,2,3,4,5}.
  - If testing return 1, the program is correct.
  - If testing return 0, the program is incorrect.

3 Program text

```c
#include <stdio.h>

int count=1,user,sum=0;
int main(){
    while(count<=5){
    printf("Input number: ");
    scanf("%d",&user);
    sum = sum+user;
    count++;
    }

    printf("Sum of numbers: %d\n", sum);

}
```

Main Program

```c
#include <stdio.h>

int count=1,user,sum=0,i;
int x[5] = {1,2,3,4,5};

int main(){
    for(i=1;i<=5;i++){
        i = count;
        while(count<=5){
            user = x[count-1];
            sum = sum+user;
            count++;
        }

    }
    printf("%d",sum == 15);
}
```

Test Program

Part 4: Terminal output is on the next page.

4 Terminal output

```
Input number: 1
Input number: 2
Input number: 3
Input number: 4
Input number: 5
Sum of numbers: 15
```

Termianl: Main program

```
1
```

Termianl: Test program

Experiment II

---

1 Problem description

- By using **do-while loop**, write a program that gets number of people that enter or exit a supermarket and shows current number of people in the supermarket. Stop the program when '0' is obtained.
- Also, verify your program.

2 Program design

- The do-while loop work as while loop but write in different prefix.
- So we set 2 variables. current[For sum total people], input[For get user input]
- The loop will break when user input 0. That mean, the condition in while is input != 0.
- For program, we will get user input.
- For verify and testing. we will use x[10] set.In x will have 1,2,3,4,5,-2,-3,6,-4,0
  - If testing return 1, the program is correct.
  - If testing return 0, the program is incorrect.

3 Program text

```c
#include <stdio.h>

int current=0,input;

int main(){
    do
    {
        printf("Insert No. people:");
        scanf("%d",&input);
        current += input;
        printf("Sum of numbers: %d\n",
            current);
    } while (input != 0);

}
```

Main Program

```c
#include <stdio.h>

int current=0,input,i;
int x[10] = {1,2,3,4,5,-2,-3,6,-4,0};

int main() {
    for (i = 0; i <= 9; i++) {
        input = x[i];
        do {
            current += input;
            input = x[++i];
        } while (input != 0);
    }
    printf("%d",current==12);
}
```

Test Program

Part 4: Terminal output is on the next page.

4 Terminal output

```
Sum of numbers: 1
Insert No. people:2
Sum of numbers: 3
Insert No. people:3
Sum of numbers: 6
Insert No. people:4
Sum of numbers: 10
Insert No. people:5
Sum of numbers: 15
Insert No. people:-2
Sum of numbers: 13
Insert No. people:-3
Sum of numbers: 10
Insert No. people:6
Sum of numbers: 16
Insert No. people:-4
Sum of numbers: 12
Insert No. people:0
Sum of numbers: 12
```

Termianl: Main program

```
1
```

Termianl: Test program

Experiment III

---

1 Problem description

- By using **for loop**, write a program that sums up value from 100 to 120. Also, find average value.
- Also, verify your program.

2 Program design

- The for loop is like while but in the limit that we can set .
- So we set 4 variables. count[For counting], sum[For sum number], i[For the for loop],
- From problem we have to start from 101 to 120 and we can step by 1. That mean, the condition in for is i=101;1¡=250;i++.
- For verify and testing. The output from sum and avg have to be 2210 and 110.0000000
  - If testing return 1, the program is correct.
  - If testing return 0, the program is incorrect.

3 Program text

```c
#include <stdio.h>

int count=0;
int sum,i;
float avg;

int main(){
    for (i=101;i<=120;i++)
    {
        printf("i=%d\n",i);
        sum += i;
        count++;
    }
    avg = sum/count;
    printf("Sum=%d Avg=%f",sum, avg);
}
```

Main Program

```c
#include <stdio.h>

int count=0;
int sum,i;
float avg;

int main(){
    for (i=101;i<=120;i++)
    {
        sum += i;
        count++;
    }
    avg = sum/count;
    printf("%d",sum==2210&&avg==110);
}
```

Test Program

Part 4: Terminal output is on the next page.

4 Terminal output

```
i=101
i=102
i=103
i=104
i=105
i=106
i=107
i=108
i=109
i=110
i=111
i=112
i=113
i=114
i=115
i=116
i=117
i=118
i=119
i=120
Sum=2210 Avg=110.000000
```

Termianl: Main program

```
1
```

Termianl: Test program

Experiment IV

---

1 Problem description

- By using **nested for loop**, write a program that shows combinations of throwing 2 dices and sum of marks is equal to 7.
- Ex. 3+4, 4+3, 2+5, 5+2,....
- Also, verify your program.

2 Program design

- The nested for loop is for loop but have other for loop in it.
- So we set 2 variables. i and j for for loop.
- From problem, we want the combination that equal to 7 that mean we have to set if-else. the condition in for is i=1;1¡=6;i++ and j=1;j¡=6;j++ and the condition in if else is i+j==7.

3 Program text

```
#include <stdio.h>

int i,j;

int main(){
    for (i=1;i<=6;i++){
        for (j=1;j<=6;j++){
            if (i+j==7)
            {
                printf("%d+%d\n",i,j);
            }

        }

    }

}
```

Part 4: Terminal output is on the next page.

4 Terminal output

```
1+6
2+5
3+4
4+3
5+2
6+1
```

End of Lab Experiment Week 4

Experiment Extra I

---

1 Problem description

- According to table below, change condition of while loop in lab4-1.c,

Q 1.1: run and record number of iteration that "Hello" is displayed on screen as well as giving the reason in the last column.

Trial No.	Condition	No. of iteration	Reason
1	true		
2	1		
3	false		
4	0		
5	'a'		
6	22.55		
7	i!=5		
8	i<8		
9	i>5 &&i<15		
10	i>5 —— i<15		

Q 1.2: Write a flowchart of trail no. 7 this program.

2 Problem program text

```c
#include <stdio.h>
#include <stdbool.h>

int main(void){
        int i=1;

        while(22.55){
            printf("Hello\n");
            i++;
        }

        return 0;
}
```

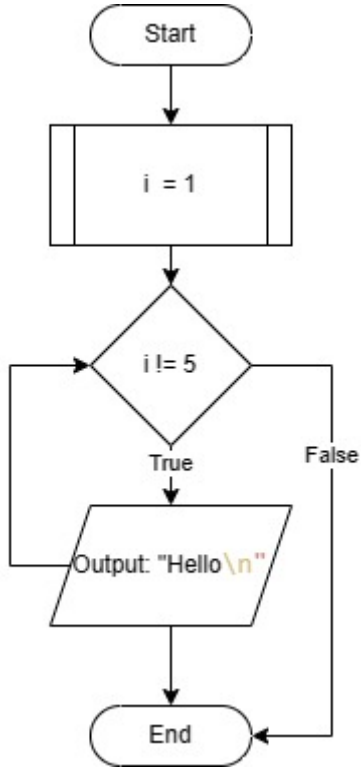
3 Answer

Ans 1.1: Table of Experiment I

Trial No.	Condition	No. of iteration	Reason
1	true	inf.	When the condition in while() is true, It will continue looping until condition is false
2	1	inf.	Same reason as No.1, 1 in boolen is true
3	false	None.	The condition in while() is false. so the loop is break
4	0	None.	Same reason as No.3, 0 in boolen is false
5	'a'	inf.	The condition in while() is 'a' which is always true
6	22.55	inf.	Same reason as No.5, The condition is 22.55 is always ture
7	i!=5	4	So the condition is when i not equal to 5 and i is 1. So, the loop will be run only 4 times.
8	i<8	7	Same as No.7, The condition is true when i is less than 8. When i is equal 8, the loop will break
9	i>5 &&i<15	None.	The condition is when i more than 5 and less than 15. i is equal to 1 so it will be false and true equal to false. So, The loop will break.
10	i>5 —— i<15	inf.	The condition is when i more than 5 or less than 15. There are no case that i is less than 5 and more than 15. So, The condition will always turn

Part 4: Flowchart design is on the next page.

4 Flowchart design



Flowchart trial No.7 [Ans 1.2]

Experiment Extra II

---

1 Problem description

- According to table below, change condition of while loop in lab4-2.c,

Q 2.1: run and record number of iteration that "Hello" is displayed on screen and record value of 'i' of each iteration in the table.

Trial No.	Exp1	Exp2	Exp3	No. of iteration	Value of variable 'i' of each iteration
1	i=0	i<10	i++		
2	i=0	i<=10	i++		
3	i=1	i<=10	i++		
4	i=1	i<=10	i+=2		
5	i=10	i>=0	i–		
6	i=10	i>=0	i-=2		

Q 2.2: Write a flowchart of trail no. 4 of this program.

Q 2.3: Explain how to modify this program in order to calculate sum and average of values from 101 to 250.

2 Problem program text

```c
#include<stdio.h>

int main(void)
{
        int i;
        int count = 0;
    int sum = 0;
        for(i=10;i>=0;i--)
    {
        printf("Hello %d\n",i);
        count++;
        sum = sum+i;
    }
    printf("%d %f", count, (float)(sum/count));

        return 0;
}
```

3 Answer

Ans 2.1: Table of Experiment II

Trial No.	Exp1	Exp2	Exp3	No. of iteration	Value of variable 'i' of each iteration
1	i=0	i<10	i++	10	Increasing in loop from 0 to 9
2	i=0	i<=10	i++	11	Increasing in loop from 0 to 10
3	i=1	i<=10	i++	10	Increasing in loop from 1 to 10
4	i=1	i<=10	i+=2	5	Increasing in loop in odd number from 1 to 9 {1,3,5,7,9}
5	i=10	i>=0	i–	11	Decreasing in loop from 10 to 0.
6	i=10	i>=0	i-=2	6	Decreasing in loop in even number from 10 to 0 {10,8,6,4,2,0}

Ans 2.3: In this case, we can modify programming in 2 ways.

Case 1: modify only for loop: In this case we can change condition to i=101;i¡=250;i++

Case 2: modify for loop to count in odd number: In this case similar like case above but we change from i++ to i+=2 and multiply be 2 in count after breaking loop for lower time and faster compiling code.

```c
#include <stdio.h>

int main(void)
{
        int i;
        int count = 0;
    int sum = 0;
        for(i=101;i<=250;i++)
    {
        printf("Hello %d\n",i);
        count++;
        sum = sum+i;
    }
    printf("%d %f", count, (float)(
        sum/count));

        return 0;
}
//150 175.000000
```

Case 1

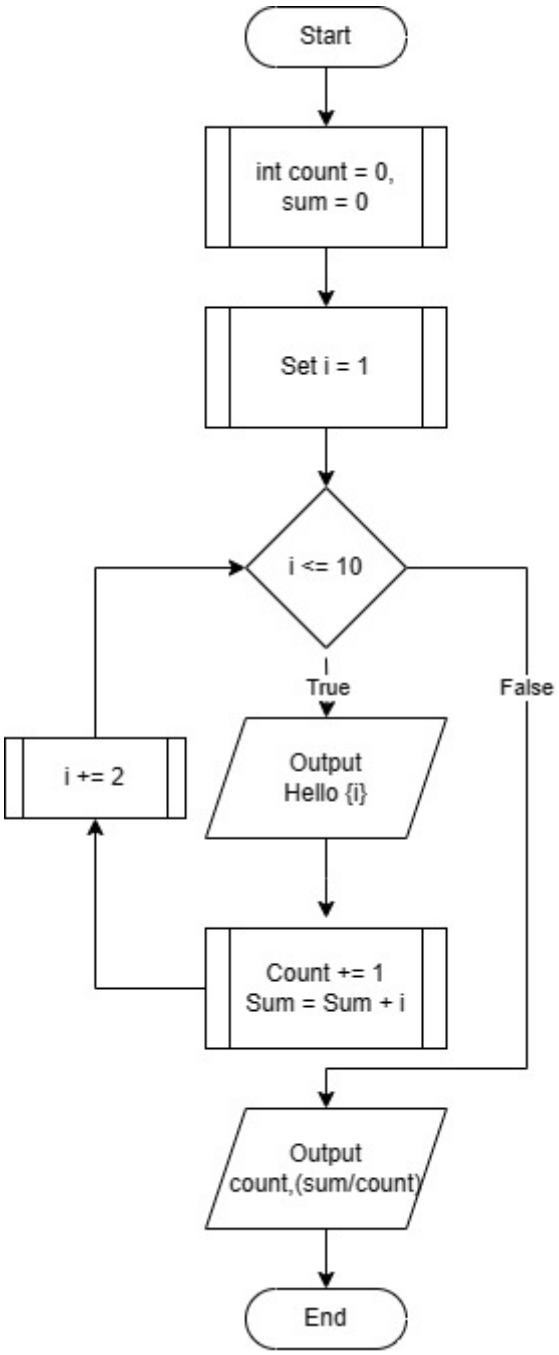
```c
#include <stdio.h>

int main(void)
{
        int i;
        int count = 0;
    int sum = 0;
        for(i=101;i<=250;i+=2)
    {
        printf("Hello %d\n",i);
        count++;
        sum = sum+i;
    }
    printf("%d %f", count*2, (float)
        (sum/count));

        return 0;
}
//150 175.000000
```

Case 2

All 2 case output same number.

Part 4: Flowchart design is on the next page.

4 Flowchart design



Flowchart trial No.4 [Ans 2.2]

Experiment Extra III

---

1 Problem description

- According to table below, change value of hidden_number in lab4-3.c,

Q 3.1: run and put in your guess as in the table. Record number of iteration that "Guess my number:" is displayed on screen as well as giving the reason.

Trial No.	hidden_number	Guess number	No. of iteration	Reason
1	5	5		
2	5	7,12,2,19,4,16,5		
3	7	7		
4	7	8,2,9,10,20,6,3,1,7		

Q 3.2: Write a flowchart of this program.

2 Problem program text

```
#include<stdio.h>

int main(void)
{
        int hidden_number;
        int user_guess;
        hidden_number = 5;
        user_guess = 0;

        while(user_guess!=hidden_number)
        {
                printf("Guess my number :");
                scanf("%d", &user_guess);
        }
        printf("Good guess !!\n");
        return 0;
}
```

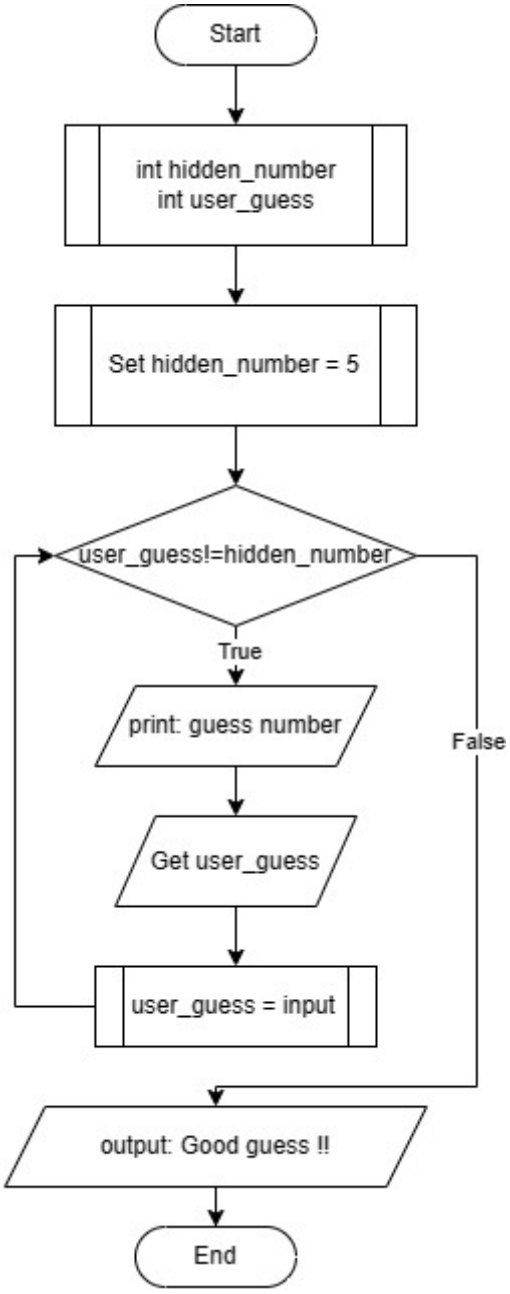
3 Answer

Ans 3.1: The condition is when user_guess not equal to hidden_number, get the number from input to user_guess and loop again.

Trial No.	hidden_number	Guess number	No. of iteration	Reason
1	5	5	1	In this case, hidden_number is not equal to user_guess. So it will loop and when user input same number (that is 5). It will break. That mean it run in loop only 1 time.
2	5	7,12,2,19,4,16,5	7	In this case, hidden_number is not equal to user_guess. So it will loop. When user input number that not equal to hidden_numeber. It will continue looping. Until user input number equal to hidden_number. The loop will break.
3	7	7	1	Same reason as No.1
4	7	8,2,9,10,20,6,3,1,7	9	Same reason as No.2

Part 4: Flowchart design is on the next page.

4 Flowchart design



Flowchart of programming process [Ans 3.2]

Experiment Extra IV

---

1 Problem description

- According to table below, change statement in lab4-4.c,

Q 4.1: run and record number of iteration and value of sum. Then, give the reason.

Trial No.	statement	No. of iteration	Sum	Reason
1	;			
2	continue;			
3	break;			

Q 4.2: Write a flowchart of trial no. 1.

2 Problem program text

```
#include<stdio.h>

int main(void)
{
        int x;
        int sum = 0 ;

        for(x=1;x<=9;x++){
                if(x==5){
                        break;            //Change this statement
                }
                else
                {
                    sum +=x;
                    printf("+%d ", x);
                }


        }
    printf("= %d", sum);

        return 0;
}
```

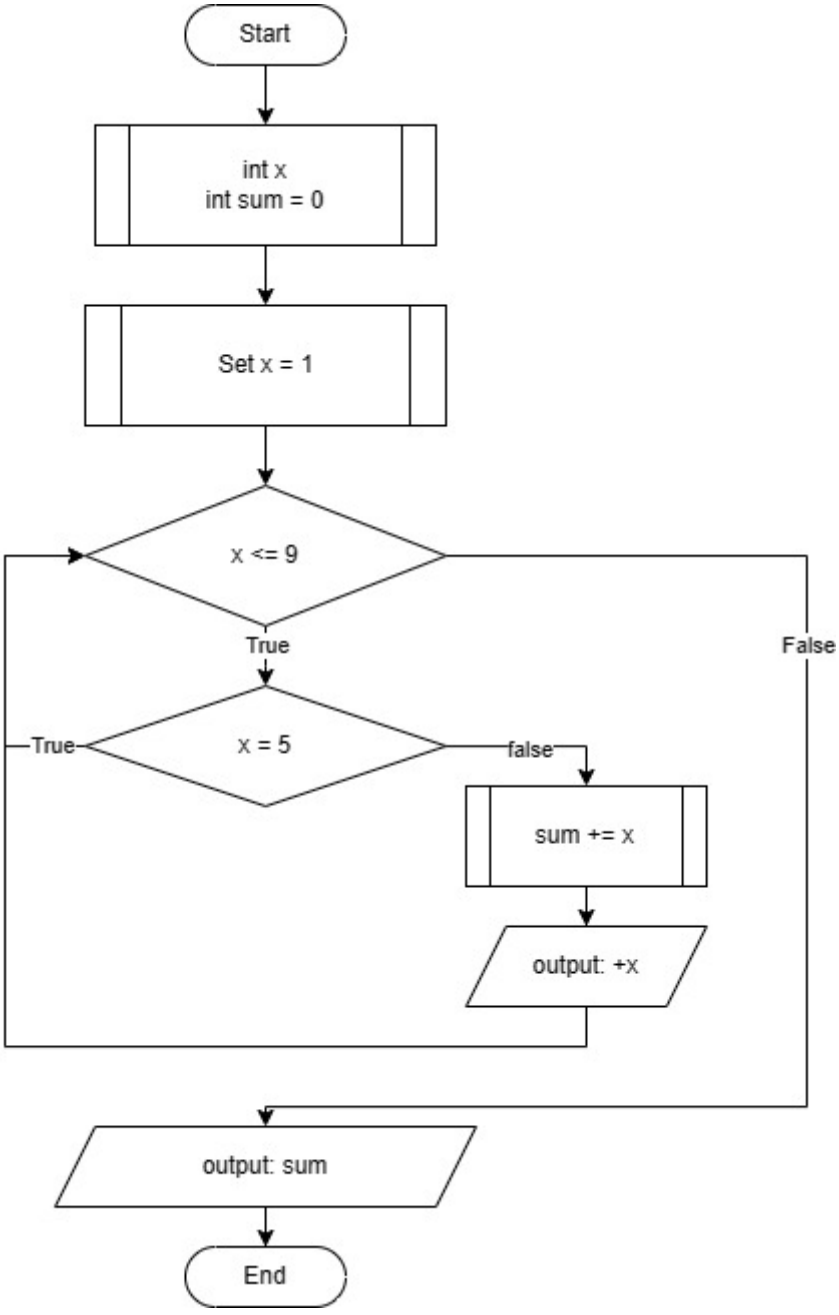
3 Answer

Ans 4.1: Table of Experiment IV

Trial No.	statement	No. of iteration	Sum	Reason
1	;	8	40	When x equal to 5, The if condition will be true. When it true, there no operation in if case. So the loop will continue.
2	continue;	8	40	Same as No.1, the continue function make loop continue looping.
3	break;	4	10	When x equal to 5, The if condition will be true. When it true, it will run break function and break (out) looping.

Part 4: Flowchart design is on the next page.

4 Flowchart design



Flowchart of trial No. 1 [Ans 4.2]

Lab Week 5

Array [I]: One Dimensional

Topic

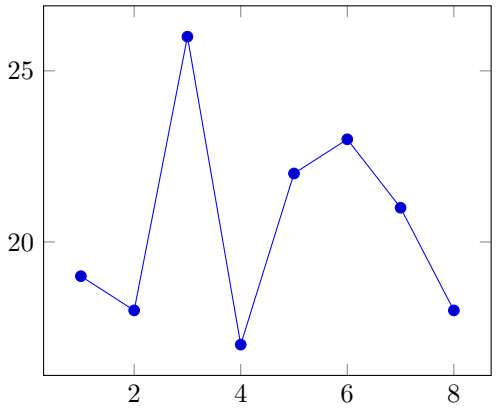
- One dimensional (1D) Array

Experiment I

---

1 Problem description

- By using **array**, write a program that stores student ages in array and finds average age according to the graph below.



2 Program design

- From the graph, we will get his value:

	Student age
1	19
2	18
3	20
4	17
5	22
6	23
7	21
8	18

- We will collect all value in array call ages[]
- After that we will find length of the array by using sizeof() function.
- For finding average, we will create function call avge() and get array and length of array.
- For calculate average or $Mean$ $[\bar{x}]$ can be calculate by $\bar{x} = \frac{\sum_{n=1}^{n} x_n}{n}$

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

int ages[] = {19, 18, 20, 17, 22, 23, 21, 18};

float avge(int list[], int length) {
    int sum = 0, i;
    float average = 0;

    for (i = 0; i < length; i++) {
        sum = sum + list[i];
    }

    average = (float)sum / length;
    return average;
}

int main() {
    printf("%f", avge(ages, sizeof(ages) / sizeof(ages[0])));
    return 0;
}
```

4 Terminal output

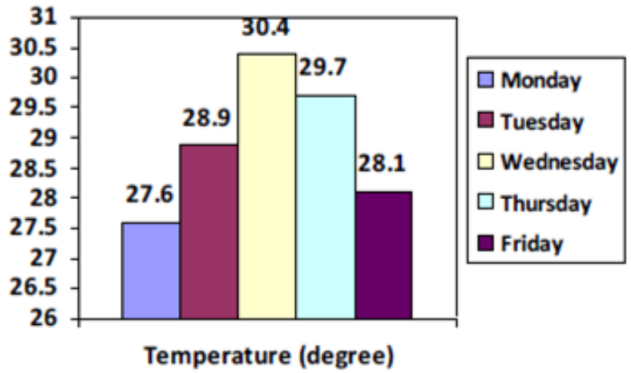
```
19.750000
```

Experiment II

---

1 Problem description

- By using **array**, write a program that identifies days that temperature is higher than 28.0 degree.



2 Program design

- From the graph, we will get his value:

Day	Temperature
Monday	27.6
Tuesday	28.9
Wednesday	30.4
Thursday	29.7
Friday	28.1

- We will collect all value in array call temp[] and day[]
- After that we will find length of the array by using sizeof() function (for limit the amount of looping).
- In this case, we want date that temperature more than 28.0.
- So, the statement we will get is $f(x_{temp}) = \begin{cases} x_{day}, & when\ x_{temp} > 28 \\ nothing, & when\ x_{temp} \leqslant 65 \end{cases}$

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

float temp[5] = {27.6,28.9,30.4,29.7,28.1};
char *day[5] = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday"};
int i;

int main(){
    for (i = 0; i <= sizeof(temp)/sizeof(temp[0]); i++){
        if (temp[i] > 28){
            printf("%s\n",day[i]);
        }
    }
}
```

4 Terminal output

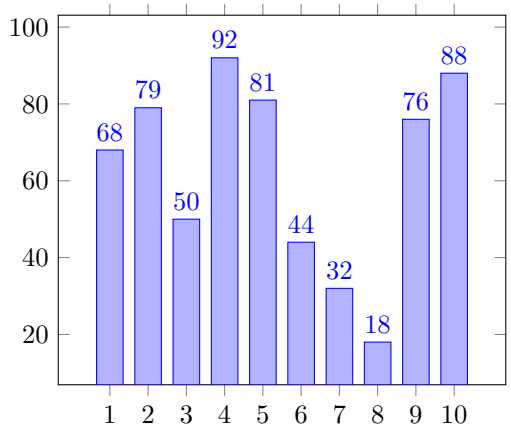
```
Tuesday
Wednesday
Thursday
Friday
```

Experiment III

---

1 Problem description

- By using **array**, Write a program that identifies day that score is maximum and minimum.



2 Program design

- From the graph, we will get his value:

	Score
1	68
2	79
3	50
4	92
5	81
6	44
7	32
8	18
9	76
10	88

- We will collect all value in array call score[]
- After that we will find length of the array by using sizeof() function (for limit the amount of looping).
- In this case, we want to find max and min.
- So, We can try to compare each other to find max and min.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

int score[10] = {68, 79, 50, 92, 81, 44, 32, 18, 75, 88};
int max, min, i;

int main() {
    max = min = score[0];
    for (i = 1; i < sizeof(score) / sizeof(score[0]); i++) {
        if (score[i] > max) {
            max = score[i];
        } else if (score[i] < min) {
            min = score[i];
        }
    }
    printf("Max: %d Min: %d\n", max, min);
}
```

4 Terminal output

```
Max: 92 Min: 18
```

End of Lab Experiment Week 5

Lab Week 6

Array [II]: Two Dimensional

Topic

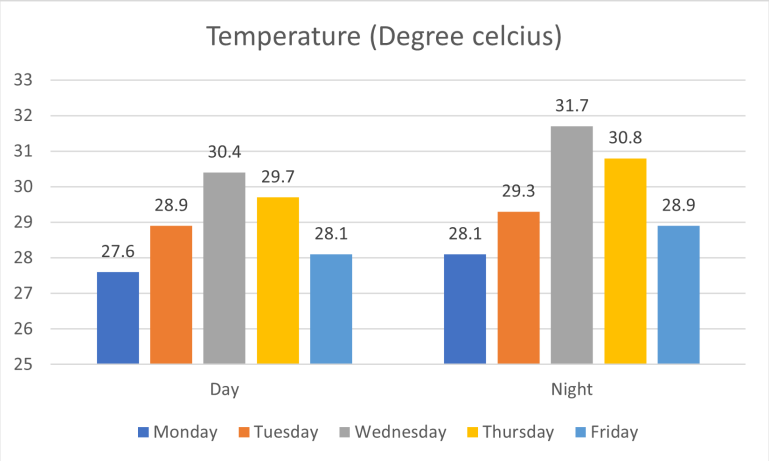
- Two dimensional (2D) Array

Experiment I

---

1 Problem description

- By using **2D array**,write a program that stores temperature value in array and finds average temperature of day and night.



2 Program design

- From the graph, we will get his value:

Day	27.6	28.9	30.4	29.7	28.1
Night	28.1	29.3	31.7	30.8	28.9

- We will collect all value in array call temp[]
- In this Experiment, we have to use 2D array. So the matrix of this array will be 2x5.
- For calculate the average of temperature in day and night, We will use function call Avgn().
- Avge() function will output average from calulating the matrix in same row by getting matrix array and lenght of column in the row that we want.
- For calculate average or $Mean$ $[\bar{x}]$ can be calculate by $\bar{x} = \frac{\sum_{n=1}^{n} x_n}{n}$

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

float temp[2][5]={{27.6,28.9,30.4,29.7,28.1},
                  {28.1,29.3,31.7,30.8,28.9}};
float avge=0;
int i,j;
int row = sizeof(temp)/sizeof(temp[0]);
int column = sizeof(temp[0])/sizeof(temp[0][0]);

float Avge(float Sheet[][5], int row, int column){
    float sum=0;
    int count=0;
    for (int i = row-1; i < row; i++){
        for (int j = 0; j < column; j++){
            // printf("%f %f\n",sum,Sheet[i][j]);
            sum = sum + Sheet[i][j];
            count++;
        }
    }
    // printf("%f %d\n", sum, count);
    avge = (float)sum/count;
    count = 0;
    return avge;
}

int main(){
    printf("column = %d, row = %d\n", column, row);
    printf("Average [Day] = %.2f\n", Avge(temp, 1, column));
    printf("Average [Night] = %.2f", Avge(temp, 2, column));
}
```

4 Terminal output

```
column = 5, row = 2
Average [Day] = 28.94
Average [Night] = 29.76
```

Part 5: Problem that occur is on the next page.

5 Problem that occur

- The problem that we get is when we compare between hand calculate and programming calculate. The decimal are not same.
- This problem could be related to how the compiler or the environment is handling floating-point precision.
- You can see the comparison between hand calculate and programming calculate and full calculate of the program.

```
column = 5, row = 2
0.000000 27.600000
27.600000 28.900000
56.500000 30.400000
86.900002 29.700001
116.600006 28.100000
144.700012 5
Average [Day] = 28.94
0.000000 28.100000
28.100000 29.299999
57.400002 31.700001
89.100006 30.799999
119.900009 28.900000
148.800003 5
Average [Night] = 29.76
```

First column [Day]

$$\bar{x} = \frac{27.6 + 28.9 + 30.4 + 29.7 + 28.1}{5}$$

$$\bar{x} = \frac{145.7}{5}$$

$$\bar{x} = 29.14$$

Second column [Night]

$$\bar{x} = \frac{28.1 + 29.3 + 31.7 + 30.8 + 28.9}{5}$$

$$\bar{x} = \frac{149.8}{5}$$

$$\bar{x} = 29.96$$

Programming output **Hand Calculate**

Experiment II

---

1 Problem description

- By using **2D array**,write a program that gets two of 3x3 matrix from user, adds them up and displays the result. Also, verify your program.

2 Program design

- In this experiment, we have to create matrix 3x3 from user input.
- So we can fix the parameter call column and row in 3.
- we can fix 2D array in 3x3 call matrix1[row][column] and matrix2[row][column]
- For getting user input to create in matrix, we use UserInputMatrix() function.
- The UserInputMatrix() function takes user input to populate a 2D matrix. The function uses nested loops to iterate over each element of the matrix, prompting the user to enter a value for each element. The matrix size is determined by the column and row parameters that are fix.
- For output matrix, we use MatrixOutput() function.
- The MatrixOutput() function prints a 2D matrix with square brackets, commas, and newlines to make it readable.
- For add 2 Matrices together, We use AddMatrix() function.
- The AddMatrix() function adds two matrices (matrix1[][] and matrix2[][]) element-wise and stores the result in another matrix.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

#define row 3
#define column 3

int matrix1[row][column] = {};
int matrix2[row][column] = {};
int result[row][column] = {};
int i, j;

void UserInputMatrix(int matrix[row][column]) {
    for (i = 0; i < row; i++) {
        for (j = 0; j < column; j++) {
            printf("Enter value: ");
            scanf("%d", &matrix[i][j]);
        }
    }
}

void MatrixOutput(int matrix[row][column]) {
    printf("[");
    for (i = 0; i < row; i++) {
        printf("[");
        for (j = 0; j < column; j++) {
            printf("%d", matrix[i][j]);
            if (j < column - 1) {
                printf(",");
            }
        }
        printf("]");
        if (i < row - 1) {
            printf("\n");
        }
    }
    printf("]");
}

void AddMatrix(int matrix1[row][column], int matrix2[row][column], int result[row][
    column]) {
    for (i = 0; i < row; i++) {
        for (j = 0; j < column; j++) {
            result[i][j] = matrix1[i][j] + matrix2[i][j];
        }
    }
}

int main() {
    printf("Enter values for matrix1:\n");
    UserInputMatrix(matrix1);
    printf("Enter values for matrix2:\n");
    UserInputMatrix(matrix2);

    printf("\nMatrix1:\n");
    MatrixOutput(matrix1);
    printf("\nMatrix2:\n");
    MatrixOutput(matrix2);

    AddMatrix(matrix1, matrix2, result);
```

```
    printf("\nSum of the matrices:\n");
    MatrixOutput(result);

    return 0;
}
```

4 Terminal output

```
Enter values for matrix1:
Enter value: 1
Enter value: 2
Enter value: 3
Enter value: 1
Enter value: 2
Enter value: 3
Enter value: 1
Enter value: 2
Enter value: 3
Enter values for matrix2:
Enter value: 1
Enter value: 2
Enter value: 3
Enter value: 1
Enter value: 2
Enter value: 3
Enter value: 1
Enter value: 2
Enter value: 3

Matrix1:
[[1,2,3]
[1,2,3]
[1,2,3]]
Matrix2:
[[1,2,3]
[1,2,3]
[1,2,3]]
Sum of the matrices:
[[2,4,6]
[2,4,6]
[2,4,6]]
```

Experiment III

---

1 Problem description

- By using **2D array**,write a program that displays total number of student who get grade A, B, C, D, and F. Also, verify your program.

	A	B	C	D	F
Math	20	15	30	2	0
English	25	10	20	10	2
Physics	15	20	10	5	3

2 Program design

- In this experiment, we have to calculate total people in that grade.
- So, we have to create matrix 3x5 call ScoreSheet[3][5] to store the value.
- We create Grade[] to set grade value(length of array will same to ScoreSheet[][] column length) and TotalScoreGrade[] to get final value.
- For counting the total number of student who get grade, we use ScoreCount() function.
- The ScoreCount() function calculates total number of student who get grade based on the input RawSheet array (ScoreSheet[][]). The function iterates through the 2D array, accumulating the count of students for each grade in the output Sheet array (TotalScoreGrade[]).
- For output the total number of student who get grade, we use ScoreOutput() function.
- The ScoreOutput() function displays the total number of students for each grade. The function takes arrays of grade labels (Grade[]) and corresponding student counts (TotalScoreGrade[]) as parameters, iterating through them to print each grade and its associated student count.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

int ScoreSheet[3][5]={{20,15,30,2,0},
                      {25,10,20,10,2},
                      {15,20,10,5,3}};
char *Grade[5] = {"A","B","C","D","F"};
int TotalScoreGrade[5]={};
int i,j;
int row = sizeof(ScoreSheet)/sizeof(ScoreSheet[0]);
int column = sizeof(ScoreSheet[0])/sizeof(ScoreSheet[0][0]);

void ScoreOutput(char *Grade[], int Sheet[], int lenght){
    for (i = 0; i < lenght; i++){
        printf("Grade %s: %d\n", Grade[i], Sheet[i]);
    }
}

int ScoreCount(int RawSheet[][5], int Sheet[], int column, int row){
    for (i = 0; i < row; i++){
        for (j = 0; j < column; j++){
            Sheet[j] = Sheet[j]+RawSheet[i][j];
        }
    }
}

int main(){
    ScoreCount(ScoreSheet,TotalScoreGrade,column,row);
    ScoreOutput(Grade,TotalScoreGrade,column);
}
```

4 Terminal output

```
Grade A: 60
Grade B: 45
Grade C: 60
Grade D: 17
Grade F: 5
```

Experiment IV

---

1 Problem description

- By using **2D array**,write a program that stores your first and last name in the array. Then, counts and displays number of character 'A' and 'a' in the array. For example,

J	o	h	n	
S	m	i	t	h

2 Program design

- In this experiment, we have to find how many upper letter and lower letter in name.
- In this time, we include string.h for easy to split text.
- So, we have to create matrix call Name and create array call User to collect user input after split.
- For collect firstname and lastname in matrix, we use StringToMatrix() function.
- The StringToMatrix() function converts the input string into a 2D matrix, where each row corresponds to a word and each column to a letter in the word. In this case is firstname and lastname.
- For counting upper and lower letter, we use CountLetterMatrix() function.
- The CountLetterMatrix() function counts the number of uppercase and lowercase letters in the matrix. The function utilizes nested loops to iterate through each element in the matrix.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#include <string.h>

#define MAXLENGHT 50

char Name[2][MAXLENGHT];
char User[MAXLENGHT];
int i, j;
int count = 0, row = 0;
int countUpper = 0, countLower = 0;

void StringToMatrix(char str[],char matrix[][MAXLENGHT]){
    for (i = 0; User[i] != '\0'; i++) {
        if (str[i] != ' ') {
            matrix[row][count] = str[i];
            count++;
        } else if (str[i] == ' ') {
            row++;
            count = 0;
        }
    }
}

void CountLetterMatrix(char matrix[][MAXLENGHT]){
    for (i = 0; i < 2; i++) {
        for (j = 0; matrix[i][j] != '\0'; j++) {
            if (matrix[i][j] >= 'A' && matrix[i][j] <= 'Z') {
                countUpper++;
            } else if (matrix[i][j] >= 'a' && matrix[i][j] <= 'z') {
                countLower++;
            }
        }
    }
}

int main() {
    printf("Enter Name: ");
    fgets(User, sizeof(User), stdin);
    User[strcspn(User, "\n")] = '\0';

    StringToMatrix(User,Name);
    CountLetterMatrix(Name);

    printf("Number of 'A' characters: %d\n", countUpper);
    printf("Number of 'a' characters: %d\n", countLower);
}
```

4 Terminal output

```
Enter Name: John Smith
Number of 'A' characters: 2
Number of 'a' characters: 7
```

End of Lab Experiment Week 6

End of Mid-term Experiment
Go on to the next page for Final-term Experiment

Lab Week 7

String

Topic

- String

Experiment I

---

1 Problem description

- Write a program that stores your name in a string and displays a welcome message.
- Ex. Welcome John Smith.

2 Program design

- So, we collect name in Name[] variable
- We can set the length in any length depend on how long is your name (In this case we don't set the length for easy to edit or change the value in variable).

3 Program text

```c
#include <stdio.h>
char Name[] = "Vivid Padungkiatsakul";

int main(){
    printf("Welcome %s", Name);
}
```

4 Terminal output

```
Welcome Vivid Padungkiatsakul
```

Experiment II

---

1 Problem description

- Write a program that gets your information by using scanf. The information are as follows:
  - Student name
  - Student surname
  - Student ID
  - Department
  - Gender
  - Age
- Then, display all information on the screen.

2 Program design

- In this case, we can do in 2D array but we will collect in own variable for easy to view and edit.
- We include string.h for using more functions to edit the string.
- For the Department, we will get this information from Student ID. In KMUTNB, the 6-8 numbers of Student ID is telling that what department that you in. For me, I in Department of **Production and Robotic Engineering**, So, the code will be **"236"** and **"216"** but for easy we will use only **"236"** to check.
- For the compare code and get department, We will use DepartmentCheck() function.
- The function work by getting the 6-8 of ID and compare to Department code. if it true, the return that department name.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#include <string.h>

char Fname[50],Sname[50],ID[14],Department[],Gender[2],Age[3];

const char *DepartmentCheck(char ID[]) {
    static char DepartmentCode[4];
    for (int i = 0; i < 3; ++i) {
        DepartmentCode[i] = ID[i + 5];
    }
    DepartmentCode[3] = '\0';

    if (strcmp(DepartmentCode, "236") == 0) {
        return "Production and Robotic Engineering";
    } else {
        return "Unknown Department";
    }
}


int main(){
    printf("Name & ID: ");
    scanf("%s %s %14s", Fname, Sname, ID);
    const *Department = DepartmentCheck(ID);
    printf("Gender [M/F] & Age: ");
    scanf("%s %s", Gender, Age);
    printf("Name: %s\nSurname: %s\nID: %s\nDepartment: %s\nGender: %s\nAge: %s",
        Fname, Sname, ID, Department, Gender, Age);
}
```

4 Terminal output

```
Name & ID: Vivid Padungkiatsakul 6601023620026
Gender [M/F] & Age: M 19
Name: Vivid
Surname: Padungkiatsakul
ID: 6601023620026
Department: Production and Robotic Engineering
Gender: M
Age: 19
```

Experiment III

---

1 Problem description

- By using array of string, write a program that gets information of 3 students from user and displays
  - Student name
  - Student surname
  - Student ID

2 Program design

- In this case, we can collect in 3D array where the plane and row is row and column of the student data and the column is how long of the string
- For getting student data, we use UserInputMatrix() function.
- The UserInputMatrix() function takes user input to populate a 3D matrix. The function uses nested loops to iterate over each element of the matrix, prompting the user to enter a value for each element. The matrix size is determined by the column and row parameters that are fix.
- For output student data, we use StdOutput() function.
- The StdOutput() function prints student data from 3D array.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#define MAXLENGHT 50

char Std[3][3][MAXLENGHT];
int plane = sizeof(Std)/sizeof(Std[0]);
int row = sizeof(Std[0])/sizeof(Std[0][0]);
int column = sizeof(Std[0][0])/sizeof(Std[0][0][0]);
int i,j,k;

void UserInputMatrix(char matrix[plane][row][column]) {
    for (i = 0; i < plane; i++) {
        printf("\nEnter details for student %d:\n", i + 1);
        printf("Enter student name: ");
        scanf("%s", matrix[i][0]);
        printf("Enter student surname: ");
        scanf("%s", matrix[i][1]);
        printf("Enter student ID: ");
        scanf("%s", matrix[i][2]);
    }
}

void StdOutput(char matrix[plane][row][column]){
    for (i = 0; i < plane; i++) {
        printf("\nDetails for student %d:\n", i + 1);
        printf("Name: %s\n", matrix[i][0]);
        printf("Surname: %s\n", matrix[i][1]);
        printf("ID: %s\n", matrix[i][2]);
    }
}

int main(){
    UserInputMatrix(Std);
    StdOutput(Std);
}
```

4 Terminal output

```
Enter details for student 1:
Enter student name: Test1
Enter student surname: STest1
Enter student ID: 111

Enter details for student 2:
Enter student name: Test2
Enter student surname: STest2
Enter student ID: 222

Enter details for student 3:
Enter student name: Test3
Enter student surname: Stest3
Enter student ID: 333

Details for student 1:
Name: Test1
Surname: STest1
ID: 111

Details for student 2:
```

```
Name: Test2
Surname: STest2
ID: 222

Details for student 3:
Name: Test3
Surname: Stest3
ID: 333
```

Experiment IV

---

1 Problem description

- Given 2 strings: "King" and "Mongkut".
- Write a program that
  - combine the strings (result should be "King Mongkut")
  - compare them.
  - copy "King" to "Mongkut".

2 Program design

- For this case, we will create the char variable for collect "King" and "Mongkut"
- For combine we will copy to other variable called CombineStr.
- In compare string 1 and 2, we will try to see is that the same and the position. So, we will compare and get the value and compare to 0.

3 Program text

```c
#include <stdio.h>
#include <string.h>

char String1[] = "King";
char String2[] = "Mongkut";

char CombineStr[20];

int main() {
    strcpy(CombineStr, String1);
    strcat(CombineStr, " ");
    strcat(CombineStr, String2);

    printf("String1[%s] + String2[%s]: %s\n", String1, String2, CombineStr);
    printf("Compare String1[%s] to String2[%s]: %d (%d)\n", String1, String2, strcmp(
        String1,String2) == 0, strcmp(String1,String2));
    printf("Copy String1[%s] to String2[%s]: %s\n", String1, String2, strcpy(String2,
        String1));
}
```

4 Terminal output

```
String1[King] + String2[Mongkut]: King Mongkut
Compare String1[King] to String2[Mongkut]: 0 (-1)
Copy String1[King] to String2[King]: King
```

End of Lab Experiment Week 7

Lab Week 8

Pointer

Topic

- Array pointer
- Variable pointer

Experiment I

---

1 Problem description

- Write a program that stores your age in a variable and declare a pointer and let the pointer points to the variable.
- Then, print address and value of the variable and of the pointer and verify that the pointer points to the variable correctly.

2 Program design

- The pointer is like container that contain variable address. When change value in pointer, the value of variable that in the pointer will change to.
- For create the pointer, we use '*' to make that variable is pointer.
- For pointing the pointer, we will use [pointer] = &[variable] to get the address of the variable.
- For output the address, we use '%p' to output the address variable and use '&' for output the address the pointer.

3 Program text

```c
#include <stdio.h>

int age = 19;
int *pointer;

int main(){
    pointer = &age;
    printf("Age (in age): %d\nAge (in pointer): %d\nAddress age: %p\nCompare adress
        between variable and pointer: %d",age,*pointer,pointer, &age==pointer);
}
```

4 Terminal output

```
Age (in age): 19
Age (in pointer): 19
Address age: 00007ff75f122010
Compare adress between variable and pointer: 1
```

Experiment II

---

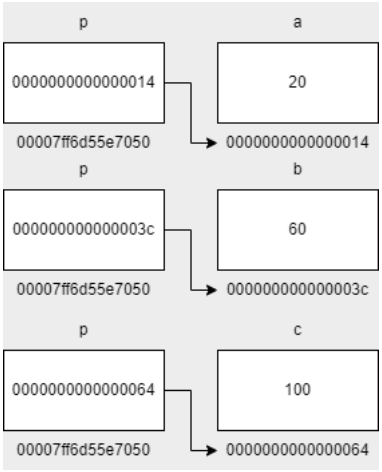
1 Problem description

- Write a program that has variables a, b, and c (all are integer) and pointer p. Then, do the followings:

  1. Let p points to a and set value of a to 20.
  2. Let p points to b and set value of b to 60.
  3. Let p points to c and set value of c to 100.
  4. Modify value of a to 50 by using pointer p.
  5. Declare pointer q and change value of c to 80 by using q.
  6. Modify value of a to 500 by using pointer q.
  7. Print value of a, b, c, p, and q and draw a figure to show the final result.

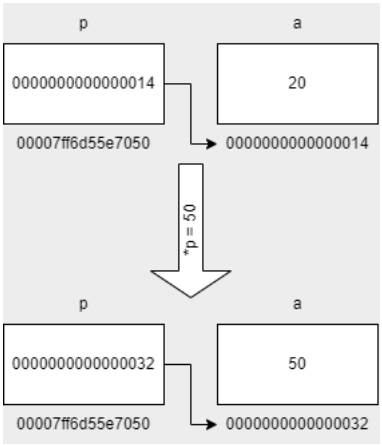
2 Program design

- In this experiment, we have to insert value into a, b and c.
- When create variable that doesn't contain the value, the address will be '0000000000000000'.
- When we point the variable and change the value, the address of that variable will change as see in the diagram below.

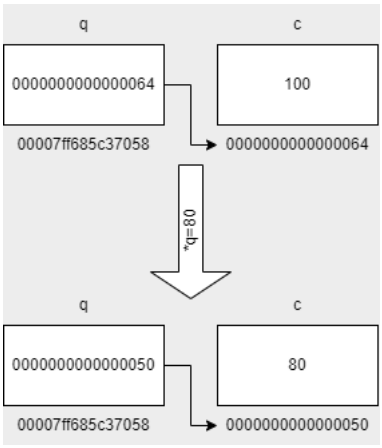


- Then we point p to a and change value to 50.

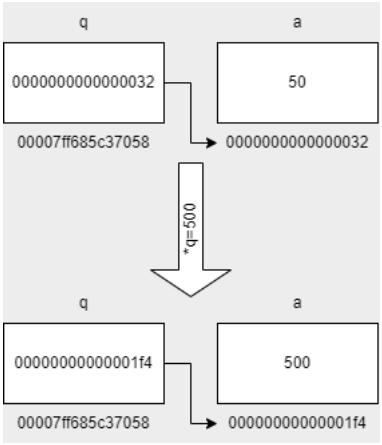
The diagram is on the next page



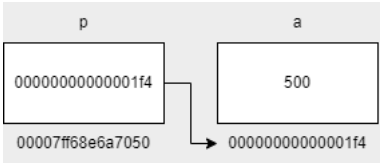
- Then we point q to c and change value to 80.



- Then we point q to a and change value to 500.



- When we check p value, the value will be same as q because they link to same address.



3 Program text

```c
#include <stdio.h>

int a, b, c;
int *p, *q;

int main(){
    printf("Adress\n");
    printf("a: %p\nb: %p\nc: %p\np: %p\nq: %p\n\n",a,b,c,&p,&q);
    p = &a;
    *p = 20;
    p = &b;
    *p = 60;
    p = &c;
    *p = 100;
    printf("(a,b,c) = (%d,%d,%d)\n",a,b,c);
    printf("Adress\n");
    printf("a: %p\nb: %p\nc: %p\np: %p\nq: %p\n\n",a,b,c,&p,&q);

    p = &a;
    *p = 50;
    q = &c;
    *q = 80;
    q = &a;
    *q = 500;

    printf("(a,b,c,p,q) = (%d,%d,%d,%d,%d)\n",a,b,c,*p,*q);
    printf("Adress\n");
    printf("a: %p\nb: %p\nc: %p\np: %p\nq: %p\n\n",a,b,c,&p,&q);
}
```

Part 4: Terminal output is on the next page.

4 Terminal output

```
Adress
a: 0000000000000000
b: 0000000000000000
c: 0000000000000000
p: 00007ff7cd1f7050
q: 00007ff7cd1f7058

(a,b,c) = (20,60,100)
Adress
a: 0000000000000014
b: 000000000000003c
c: 0000000000000064
p: 00007ff7cd1f7050
q: 00007ff7cd1f7058

(a,b,c,p,q) = (500,60,80,500,500)
Adress
a: 00000000000001f4
b: 000000000000003c
c: 0000000000000050
p: 00007ff7cd1f7050
q: 00007ff7cd1f7058
```

Experiment III

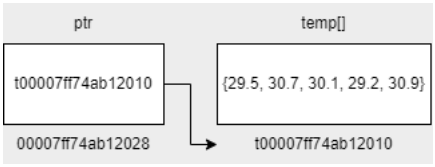
---

1 Problem description

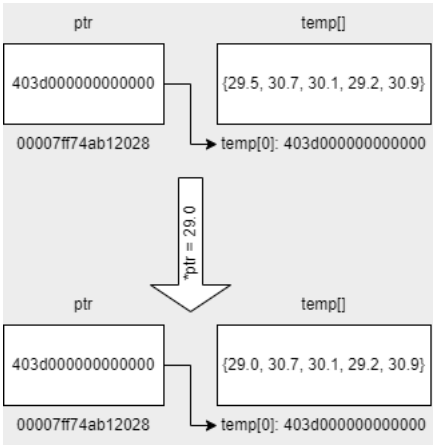
- Write a program that store temperature of 5 days (29.5, 30.7, 30.1, 29.2, 30.9). Declare a pointer and do the followings:
  1. Let the pointer points to the first value in the array.
  2. Modify the first temperature value from 29.5 to 29.0 by using the pointer.
  3. Modify the third temperature value from 30.1 to 30.0 by using the pointer and without moving the pointer.
  4. Move the pointer to the last value in the array
  5. Modify the fourth temperature value from 29.2 to 29.3.
  6. Print the array and the pointer value and draw a figure to show the final result.

2 Program design

- In this experiment, we can use pointer to point the index of the array.
- When we point the pointer to array, it will point to index 0 first. (In diagram will show pointer connect to array address)

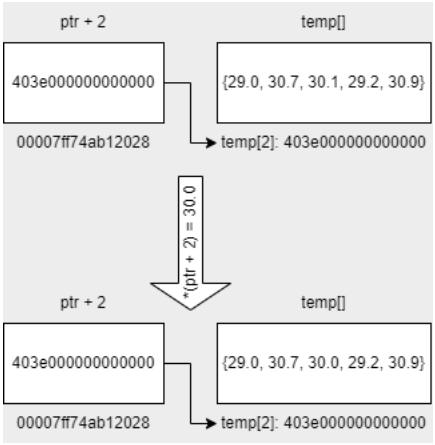


- Then we point ptr to index 0 and change value to 29.0.

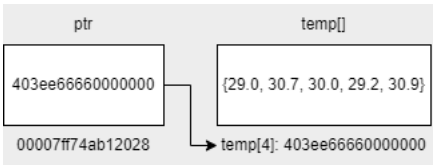


Part 2: Program design (continue) is on the next page

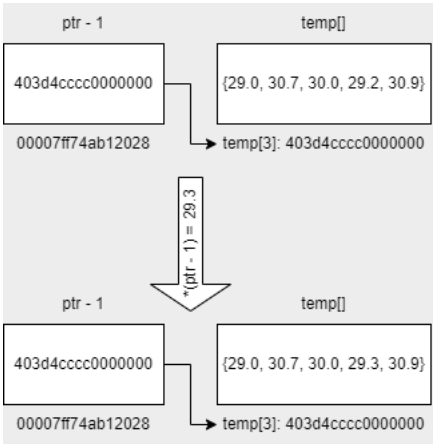
- Then we point ptr to index 2 by + index to ptr and change value to 30.0.



- Then we point ptr to index 4 aka. last value of the array.



- Then we point ptr to index 3 by -index of ptr by 1 and change value to 29.3



Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

const int length = 5;
float temp[] = {29.5, 30.7, 30.1, 29.2, 30.9};
float *ptr  = temp;
int i;

void PrintList(float list[], int length){
    for (i = 0; i < length; i++){
        printf("%.1f\n",list[i]);
    }
}

void AddressListIndex(float list[],int length){
    for (i = 0; i < length; i++){
        printf("temp[%d]: %p\n",i,list[i]);
    }
}

int main(){
    *ptr = 29.0;
    *(ptr + 2) = 30.0;
    ptr = &temp[4];
    *(ptr - 1) = 29.3;
    PrintList(temp,length);
    printf("Pointer value: %.1f\n", *ptr);
    printf("Adress\n");
    printf("temp[]: %p\n",temp);
    AddressListIndex(temp,length);
    printf("ptr: %p",&ptr);
}
```

4 Terminal output

```
29.0
30.7
30.0
29.3
30.9
Pointer value: 30.9
Adress
temp[]: 00007ff7614c2010
temp[0]: 403d000000000000
temp[1]: 403eb33340000000
temp[2]: 403e000000000000
temp[3]: 403d4cccc0000000
temp[4]: 403ee66660000000
ptr: 00007ff7614c2028
```

Experiment IV

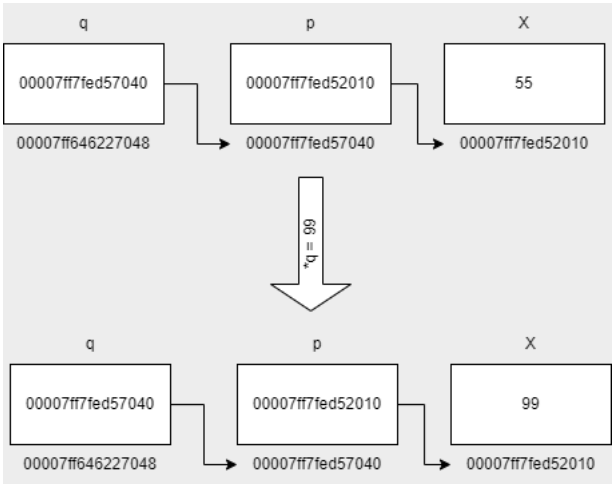
---

1 Problem description

- Write a program that store 55 in variable x and do the followings:
  1. Let pointer p points to x.
  2. Let pointer q points to p.
  3. Modify value of x to 99 by using pointer q.
  4. Move the pointer to the last value in the array
  5. Modify the fourth temperature value from 29.2 to 29.3.
  6. Print value of variables and draw a figure to verify your program.

2 Program design

- In this experiment, there are pointer and double pointer.
- For creating double pointer, we use '**' instead of using '*'.
- For point double pointer to pointer, we will use same method then we point the pointer to variable.
- When we want to change value from double pointer, we will use '**' instead of using '*'.
- So, we point double pointer to pointer to x and change value to 99.



Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

int x=55;
int *p,**q;

int main(){
    p = &x;
    q = &p;
    **q = 99;
    printf("Address\nx: %p\np: %p\nq: %p\n",p,&p,&q);
    printf("x = %d",x);
}
```

4 Terminal output

```
Address
x: 00007ff7804a2010
p: 00007ff7804a7040
q: 00007ff7804a7048
x = 99
```

End of Lab Experiment Week 8

Lab Week 9

Function[I]: Void & Value

Topic

- Functions
- Define Function
- Pass by value

Experiment I

---

1 Problem description

- Write a function that shows a box.
- void drawBox(void)

```
************************
*                      *
*                      *
************************
```

- In function main, call function drawFigure() 3 times.

2 Program design

- In this experiment, we have to create function called "drawFigure" to draw 24x4 box.
- So first, we have to create the top off the box by loop print '*' 24 times.
- Then, for the middle part that have a hole by double loop. The first for loop for the height of the middle part and other loop is for the width of the middle part.
- For the bottom part, we will do the same like the top part.
- For main function, we will loop the drawFigure function 3 times.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
int i;
int width = 24, height = 4;

void drawBox(){
    //Top part
    for (int i = 0; i < width; i++) {
        printf("*");
    }
    printf("\n");

    //Middle part
    for (int i = 0; i < height - 2; i++) {
        printf("*");
        for (int j = 0; j < width - 2; j++) {
            printf(" ");
        }
        printf("*\n");
    }

    //Bottom part
    for (int i = 0; i < width; i++) {
        printf("*");
    }
    printf("\n");
}

void main(){
    for (i = 0; i < 3; i++){
        drawBox();
    }
}
```

4 Terminal output

```
************************
*                      *
*                      *
************************
************************
*                      *
*                      *
************************
************************
*                      *
*                      *
************************
```

Experiment II

---

1 Problem description

- Write a function that calculates area of a triangle.
- void calAreaTriangle(float width, float height)
- In function main, call function calArea Triangle() and verify the function.

2 Program design

- For this problem, the function have input 2 variable, width and height, and the function is void. That mean, it can't return the value out of the function.
- For calculate the area of triangle(A), we will use:

$$A = \frac{1}{2} \times w \times h$$

- for testing and compare to hand calculation, we will set width and height = 1.

3 Equation solve

First:

$$Let\ w\ and\ h\ = 1$$

Then insert into formula:

$$\begin{aligned} A &= \frac{1}{2} \times w \times h \\ &= \frac{1}{2} \times 1 \times 1 \\ &= \frac{1}{2} = 0.5 \end{aligned}$$

4 Program text

```c
#include <stdio.h>

void calAreaTriangle(float width, float height){
    float area = 0.5*width*height;
    printf("%f",area);
}

int main(){
    calAreaTriangle(1,1);
}
```

Part 5: Terminal output is on the next page.

5 Terminal output

```
0.500000
```

Experiment III

---

1 Problem description

- Write a function that returns a grade.
- char grading(int score)

Score	Grade
85-100	A
70-84	B
55-69	C
0-54	F

- In function main, call function grading() and verify the function.

2 Program design

- The function will be [Note: For Invalid (I) is for double check if user input more than 100 or lower than 0]:

$$grading(x_{score}) = \begin{cases} I, & when\ x_{score} > 100\ or\ x_{score} < 0 \\ A, & when\ 85 \leq x_{score} \leq 100 \\ B, & when\ 70 \leq x_{score} \leq 84 \\ C, & when\ 55 \leq x_{score} \leq 69 \\ F, & when\ x_{score} < 54 \end{cases}$$

- In this case we have to use if condition to split case.
- For the function, we return the character from the function.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
#include <stdlib.h>

char grading(int score){
    if (score > 100 || score < 0) {
        return 'I';
    } else if (score >= 85) {
        return 'A';
    } else if (score >= 70) {
        return 'B';
    } else if (score >= 55) {
        return 'C';
    } else {
        return 'F';
    }
}

int main(){
    printf("%c\n",grading(101));
    printf("%c\n",grading(rand() % 16 + 85));
    printf("%c\n",grading(rand() % 15 + 70));
    printf("%c\n",grading(rand() % 10 + 55));
    printf("%c\n",grading(rand() % 55));
    printf("%c\n",grading(-1));
}
```

4 Terminal output

```
I
A
B
C
F
I
```

Experiment IV

---

1 Problem description

- Write a function that gets weight of user.
- float getWeight(void)
- In function main, call function getWeight() and verify the function.

2 Program design

- In this problem we will get the value from user input that in the function.
- For testing, we will print the value from the function return.

3 Program text

```c
#include <stdio.h>

float getWeight(){
    float weight;
    scanf("%f", &weight);
    return weight;
}

int main(){
    printf("%.2f",getWeight());
}
```

4 Terminal output

```
55.2
55.20
```

End of Lab Experiment Week 9

Lab Week 10

Function[II]: Array & Pointer

Topic

- Function with Array
- Pass by reference

Experiment I

---

1 Problem description

- Write a function that shows a message if the temperature is higher than 30.0 degrees or not. The function gets a temperature value of an array {29.5, 30.7, 30.1, 29.2, 30.9} in function main.
- void IsTempHigh(float temperature)
- **Note**: Pass a cell of array into the function.

2 Program design

- For this experiment, we have to create function called "IsTempHigh()" and get cell value from array.
- For the function, we use if statement to split between more or equal to 30 and lower than 30
- The function will be:

$$IsTempHigh(x_{temp}) = \begin{cases} x_{temp} >= 30 & High \\ x_{temp} < 30 & NotHigh \end{cases}$$

3 Program text

```c
#include <stdio.h>
float Temp[] = {29.5, 30.7, 30.1, 29.2, 30.9};
int lenght = sizeof(Temp)/sizeof(Temp[0]);
int i;

void IsTempHigh(float temperature){
    if (temperature >= 30){
        printf("Temp: %.1f is High\n",temperature);
    } else {
        printf("Temp: %.1f is not High\n",temperature);
    }
}

int main(){
    for (i = 0; i < lenght; i++){
        IsTempHigh(Temp[i]);
    }
}
```

4 Terminal output

```
Temp: 29.5 is not High
Temp: 30.7 is High
Temp: 30.1 is High
Temp: 29.2 is not High
Temp: 30.9 is High
```

Experiment II

---

1 Problem description

- Write a function that gets an array {29.5, 30.7, 30.1, 29.2, 30.9} as input and returns average temperature. In function main, call the function and verify the function.
- float calAverageTemp(float t[])
- **Note**: Pass the whole array into the function.

2 Program design

- For this experiment, we have to create function called "calAverageTemp()" and get cell value from array by insert whole array.
- The function will sum all value of the array and calculate the average.
- For calculate the average, we can calculate by:

$$\bar{x} = \frac{\sum_{n=1}^{n} x_n}{n}$$

3 Program text

```c
#include <stdio.h>
float Temp[] = {29.5, 30.7, 30.1, 29.2, 30.9};
int lenght = sizeof(Temp)/sizeof(Temp[0]);
int i;

float calAverageTemp(float t[]){
    float SumFunction = 0, Avge;
    for (i = 0; i < lenght; i++){
        SumFunction = SumFunction + t[i];
    }
    Avge = SumFunction/lenght;
    return Avge;
}

int main(){
    printf("lenght: %d\nAverage temp: %f",lenght,calAverageTemp(Temp));
}
```

4 Terminal output

```
lenght: 5
Average temp: 30.079998
```

Experiment III

---

1 Problem description

- Write a function that increments value of a variable (declared in function main) by one every time that the function is called. In function main, call the function 10 times and verify the function.
- void count(int *p)
- **Note**: Pass address of a variable into the function.

2 Program design

- For this experiment, we have to create the function called "count" that get address from pointer and add the value of that variable of the address.
- In this case, we can not have the return value because when we add the value of the pointer, the variable of that address will add too.
- For calling function 10 times, we use the loop of easy to do the same pattern.

3 Program text

```c
#include <stdio.h>

void count(int *p){
    (*p)++;
}

int main(){
    int var = 0, i;
    printf("Var: %d\n",var);
    for(i = 0; i < 10 ; i++){
        count(&var);
    }
    printf("Var: %d", var);
}
```

4 Terminal output

```
Var: 0
Var: 10
```

Experiment IV

---

1 Problem description

- Write a function that calculate volume of cylinder.
- void calCylinderVol(float r, float h, float *v)
- In function main, call the function and verify the function.

2 Program design

- For this experiment, we have to return the value after calculation but the problem fix the function to void. So, we have to use pointer to transfer the value.
- For calculate the area of cylinder (A), we can calculate by:

$$A = \Pi r^2 h$$

- For testing and compare to hand calculation, we set r and h equal to 1.

3 Equation solve

First:

$$Let\ r\ and\ h = 1$$

Then insert into equation:

$$\begin{aligned} A &= \Pi r^2 h \\ &= \Pi \times 1^2 \times 1 \\ A &= \Pi \approx 3.141593 \end{aligned}$$

4 Program text

```c
#include <stdio.h>
#define M_PI (3.141592653589793)

void calCylinderVol(float r, float h, float *v){
    *v = M_PI * r * r * h ;
}

int main(){
    float Radiun = 1, Heigh = 1, Volumn;
    calCylinderVol(Radiun,Heigh,&Volumn);
    printf("Volumn of Cylender (R: %f, D: %f): %f",Radiun,Heigh,Volumn);
}
```

Part 5: Terminal output is on the next page.

5 Terminal output

```
Volumn of Cylender (R: 1.000000, D: 1.000000): 3.141593
```

End of Lab Experiment Week 10

Lab Week 11

Enumerator & Structure

Topic

- Enumerator
- Structure
- Structure of function (Typedef)
