EXPERIMENT7: STRUCTURES AND UNION

Experiment 1: Complex Number using Structure

Aim

To perform reading, writing, addition, and subtraction of two complex numbers using structures and functions.

Algorithm

1. Define structure with real and imaginary parts
2. Read two complex numbers
3. Add and subtract them
4. Display results

Pseudocode

```
STRUCT Complex
    real, imag
END STRUCT
```

```
FUNCTION readComplex()
FUNCTION printComplex()
FUNCTION addComplex()
FUNCTION subComplex()
```

Flowchart (ASCII)

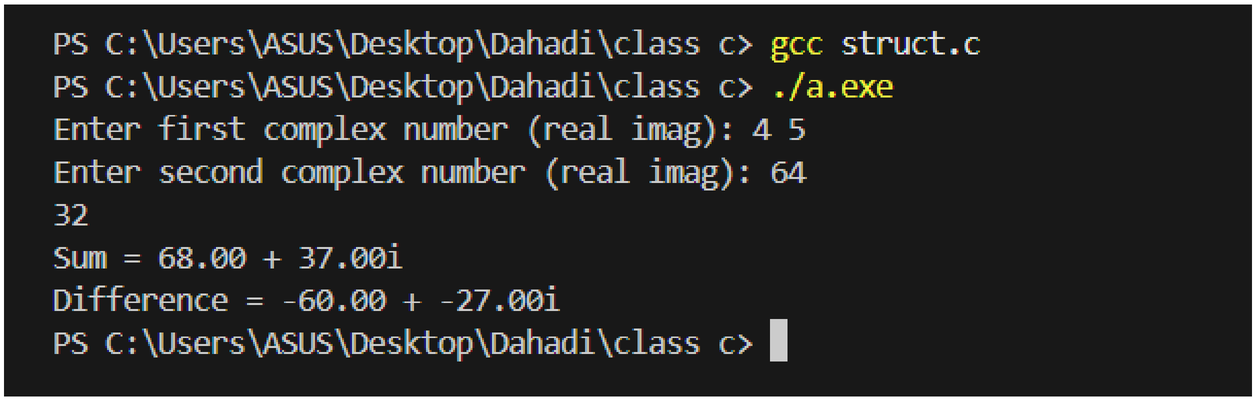
```
Start
   ↓
Read Complex Numbers
   ↓
Add / Subtract
   ↓
Display Result
   ↓
 End
```

C Program

```
#include <stdio.h>

struct Complex {
    float real, imag;
};

struct Complex readComplex() {
    struct Complex c;
scanf("%f %f", &c.real, &c.imag);
    return c;
}

void printComplex(struct Complex c) {
    printf("%.2f + %.2fi\n", c.real, c.imag);
```

```c
}

struct Complex add(struct Complex a, struct Complex b) {
    struct Complex c;
    c.real = a.real + b.real;
    c.imag = a.imag + b.imag;
    return c;
}

struct Complex sub(struct Complex a, struct Complex b) {
    struct Complex c;
    c.real = a.real - b.real;
    c.imag = a.imag - b.imag;
    return c;
}

int main() {
    struct Complex c1, c2, sum, diff;

    printf("Enter first complex number (real imag): ");
    c1 = readComplex();

    printf("Enter second complex number (real imag): ");
    c2 = readComplex();

    sum = add(c1, c2);
    diff = sub(c1, c2);

    printf("Sum = ");
    printComplex(sum);

    printf("Difference = ");
    printComplex(diff);

    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc struct.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
Enter first complex number (real imag): 4 5
Enter second complex number (real imag): 64
32
Sum = 68.00 + 37.00i
Difference = -60.00 + -27.00i
PS C:\Users\ASUS\Desktop\Dahadi\class c>
```

Conclusion

Structures and functions simplify complex number operations.

Experiment 2: Employee Salary Using Structure

Aim

To calculate and display employee gross salary using structure.

Algorithm

1. Read name and basic pay
2. Calculate DA = 10% of basic
3. Gross = Basic + DA
4. Print name and gross salary

Pseudocode

```
READ name, basic
DA = basic * 0.1
gross = basic + DA
DISPLAY name, gross
```

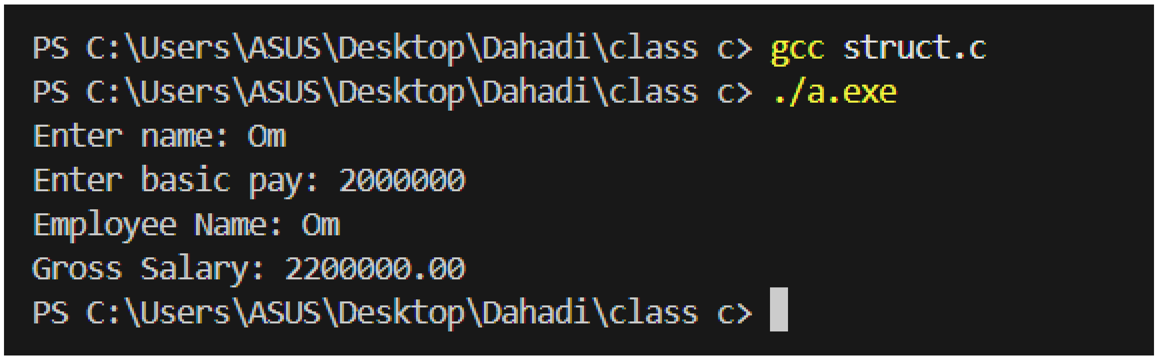
Flowchart (ASCII)

```
Start
   ↓
Read Data
   ↓
Calculate DA & Gross
   ↓
Display
   ↓
 End
```

C Program

```c
#include <stdio.h>

struct Employee {
    char name[30];
    float basic, da, gross;
};

int main() {
    struct Employee e;
    printf("Enter name: ");
    scanf("%s", e.name);

    printf("Enter basic pay: ");
    scanf("%f", &e.basic);

    e.da = e.basic * 0.10;
    e.gross = e.basic + e.da;

    printf("Employee Name: %s\n", e.name);
    printf("Gross Salary: %.2f\n", e.gross);

    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc struct.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
Enter name: Om
Enter basic pay: 2000000
Employee Name: Om
Gross Salary: 2200000.00
PS C:\Users\ASUS\Desktop\Dahadi\class c> 
```

Conclusion

Employee salary calculation was successfully implemented using structures.

Experiment 3: Book Structure Passed to Function

Aim

To pass a structure to a function and display book details.

Algorithm

1. Read book details
2. Pass structure to function
3. Print details

Pseudocode

```
STRUCT Book
READ book details
CALL printBook(book)
```

Flowchart (ASCII)

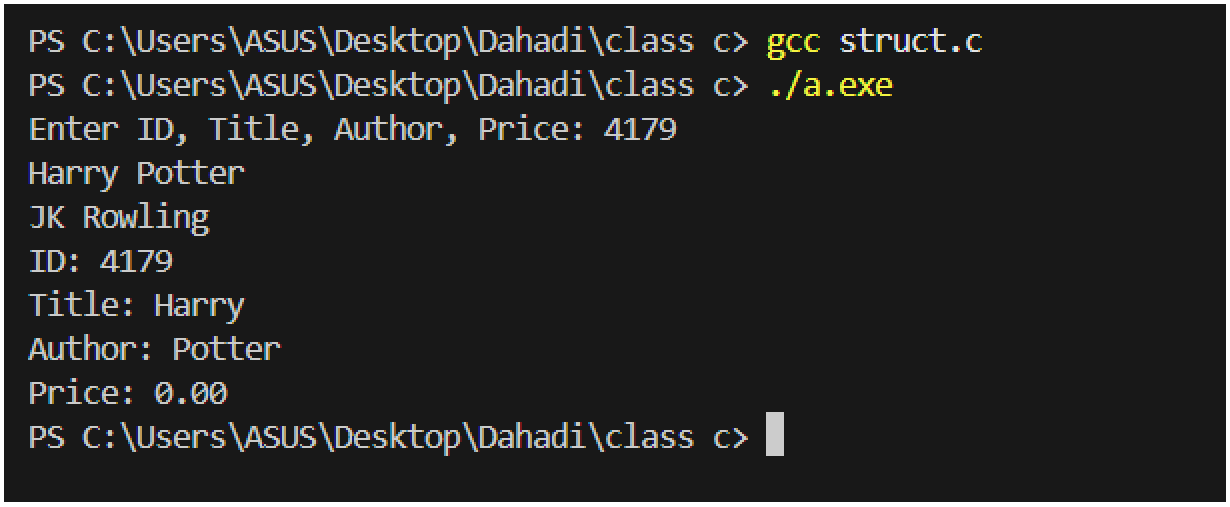
```
Start
   ↓
Read Book Data
   ↓
Pass to Function
   ↓
```

```
Display Details
   ↓
 End
```

C Program

```c
#include <stdio.h>

struct Book {
    int id;
    char title[30];
    char author[30];
    float price;
};

void printBook(struct Book b) {
    printf("ID: %d\nTitle: %s\nAuthor: %s\nPrice: %.2f\n",
           b.id, b.title, b.author, b.price);
}

int main() {
    struct Book b;

    printf("Enter ID, Title, Author, Price: ");
    scanf("%d %s %s %f", &b.id, b.title, b.author, &b.price);

    printBook(b);
    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc struct.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
Enter ID, Title, Author, Price: 4179
Harry Potter
JK Rowling
ID: 4179
Title: Harry
Author: Potter
Price: 0.00
PS C:\Users\ASUS\Desktop\Dahadi\class c>
```

Conclusion

Structures can be easily passed to functions for modular programming.

Experiment 4: Union for Address Storage

Aim

To demonstrate the use of union for storing address details.

Algorithm

1. Define union with address fields
2. Read present address
3. Display it

Pseudocode

```
UNION Address
READ address
DISPLAY address
```

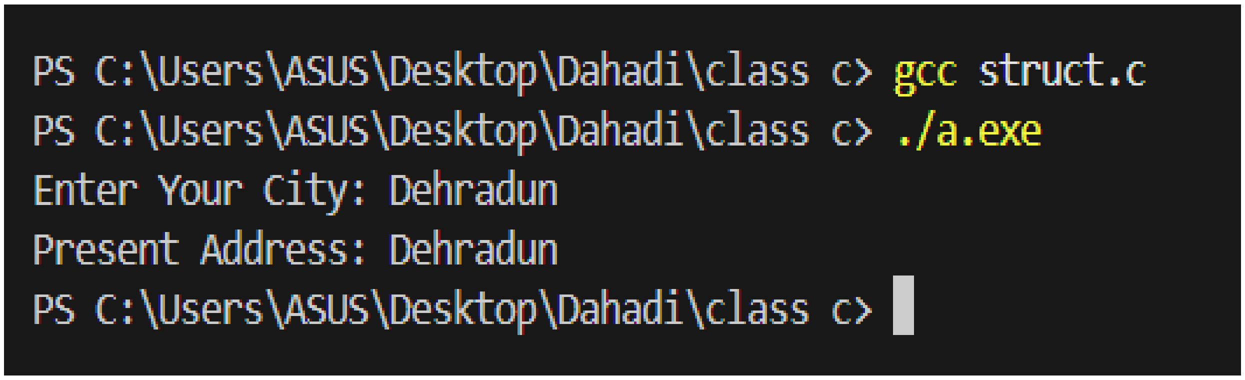
Flowchart (ASCII)

```
Start
   ↓
Input Address
   ↓
Display Address
   ↓
 End
```

C Program

```c
#include <stdio.h>

union Address {
    char home[50];
    char hostel[50];
    char city[20];
};

int main() {
    union Address addr;

    printf("Enter Your City: ");
    scanf("%s", addr.city);
    printf("Present Address: %s\n", addr.city);

    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc struct.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
Enter Your City: Dehradun
Present Address: Dehradun
PS C:\Users\ASUS\Desktop\Dahadi\class c>
```

Conclusion

Union efficiently stores different data types using shared memory.