Lab Sheet 4

1. WAP to get your name, address and display using unformatted I/O.

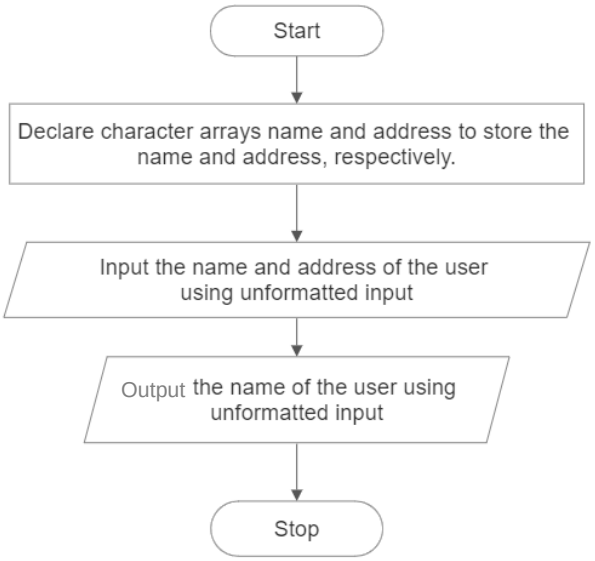
➢ Objective

The objective of this program is to input the name and address of the user and display them using unformatted I/O. Unformatted I/O allows reading and writing raw data without any specific formatting.

➢ Algorithm

i. Start.
ii. Declare character arrays name and address to store the name and address, respectively.
iii. Input the name of the user using unformatted input.
iv. Input the address from the user using unformatted input.
v. Output the name and address using unformatted output.
vi. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( ){
   char name[100], address[100];
   puts("Enter your name: ");
   gets(name);
   puts("Enter your address: ");
   gets(address);
   puts("\nYour details:\n");
   puts("Name: ");
   puts(name);
   puts("Address: ");
   puts(address);
   return 0;
}
```

➢ Output

```
Enter your name: Tilak Thapa
Enter your address: Tulsipur – 4, Dang

Your details:
Name: Tilak Thapa
Address: Tulsipur – 4, Dang
```

➢ Discussion and Conclusion

This program allows the user to input their name and address using unformatted I/O, specifically the gets() function. Unformatted I/O allows reading and writing raw data without any specific formatting.

The program uses character arrays to store the name and address provided by the user. It prompts the user to enter their name and address, reads them using gets(), and then outputs the details using unformatted output with the puts() function. The code effectively demonstrates the use of unformatted I/O in C programming for reading and writing character data. It provides a simple and straightforward solution to input and display name and address using unformatted I/O.

In conclusion, this program successfully achieves its objective of obtaining and displaying the user's name and address using unformatted I/O. It showcases the usage of unformatted input and output functions in C programming.

2. WAP to get a character form the user using unformatted I/O and display the ASCII value of the entered character.

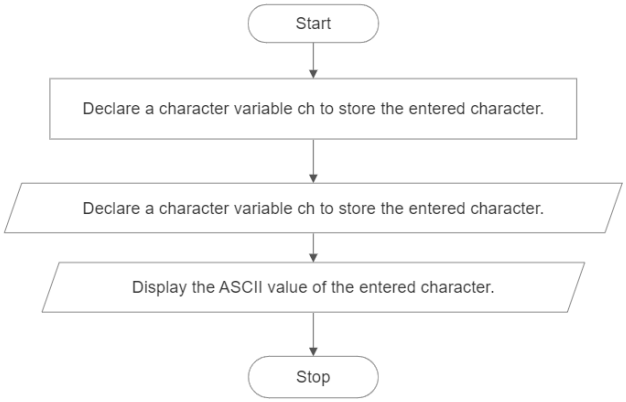
➢ Objective

The objective of this program is to input a character from the user using unformatted I/O and display the ASCII value of the entered character.

➢ Algorithm

i. Start.
ii. Declare a character variable ch to store the entered character.
iii. Prompt the user to enter a character using unformatted input.
iv. Read the character using unformatted input.
v. Display the ASCII value of the entered character.
vi. Stop

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main() {
   char ch;
   puts("Enter a character: ");
   ch = getch();
   puts("ASCII value of the entered character is: %d\n", ch);
   return 0;
}
```

➢ Output

Enter a character: A
ASCII value of the entered character is: 65

➢ Discussion and Conclusion

This program uses unformatted I/O to input a character from the user and then displays the ASCII value of the entered character.

3. WAP to display the output as [take a=15, b=20.43, c=35] as given.

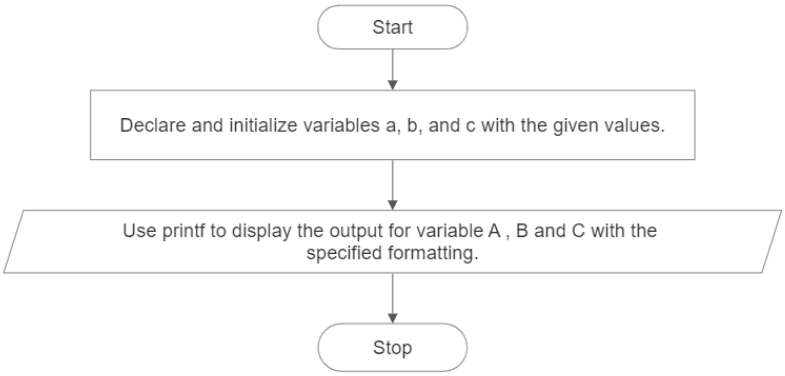
➢ Objective

The objective of this program is to display the output as specified, with the given values of variables a, b, and c, using formatted I/O.

➢ Algorithm

i. Start.
ii. Declare and initialize variables a, b, and c with the given values.
iii. Use printf to display the output for variable A , B and C with the specified formatting.
iv. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   int a = 15;
   float b = 20.43;
   int c = 35;

   printf("A = %6d|%-6d|%6d|%-6d|%6d|%-6d|%6d|%-6d|%6d|%-6d|", a, a, a, a, a, a, a, a, a, a);
   printf("\nB = %6.2f|%-6.2f|%6.2f|%-6.2f|%6.2f|%-6.2f|%6.2f|%-6.2f|%6.2f|%-6.2f|", b, b, b, b, b, b, b, b, b, b);
   printf("\nC = %6d|%-6d|%6d|%-6d|%6d|%-6d|%6d|%-6d|%6d|%-6d|", c, c, c, c, c, c, c, c, c, c);
   return 0;
}
```

➢ Output

```
A =     15|15     |     15|15     |     15|15     |     15|15     |   15|15       |
B =  20.43|20.43 | 20.43|20.43 | 20.43|20.43 | 20.43|20.43 | 20.43|20.43 |
C =     35|35     |     35|35     |     35|35     |     35|35     |   35|35       |
```

➢ Discussion and Conclusion

This program successfully displays the output as specified in the objective using formatted I/O with the specified formatting. It initializes the variables a, b, and c with the given values and then uses printf to display them in the desired format.The code uses format specifiers such as %d for integers (a and c) and %6d for right-aligned integers, %f for floating-point numbers (b) and %6.2f for right-aligned floating-point numbers with two decimal places. The - flag is used to left-align the values.

4. WAP to display the output as below using formatted I/O [take char a[]="I Love Nepal"].

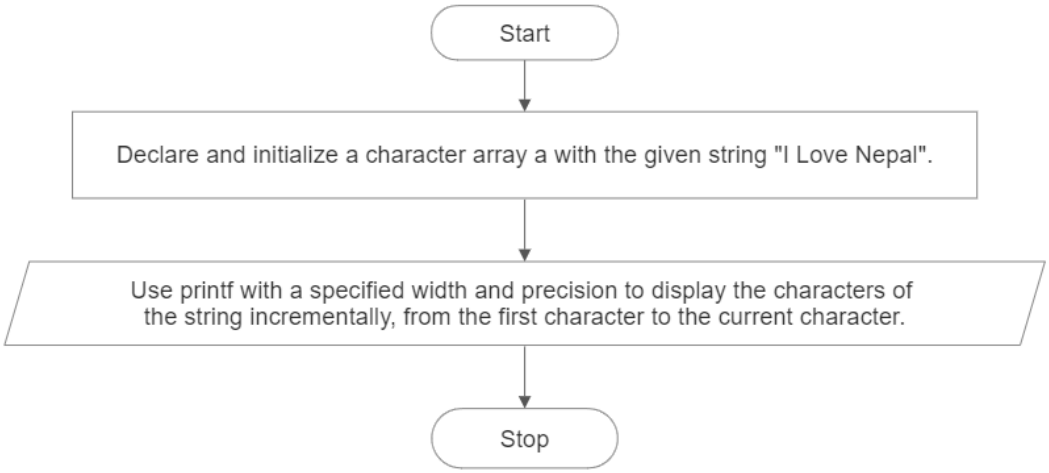
➢ Objective

The objective of this program is to display the output as specified, which is a pyramid of characters obtained from the given string "I Love Nepal" using formatted I/O.

➢ Algorithm

i. Start
ii. Declare and initialize a character array a with the given string "I Love Nepal".
iii. Use printf with a specified width and precision to display the characters of the string incrementally, from the first character to the current character.
iv. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( ) {
   char a[] = "I Love Nepal";
   printf("%1.1s\n", a);
   printf("%1.3s\n", a);
   printf("%1.4s\n", a);
   printf("%1.5s\n", a);
   printf("%1.7s\n", a);
   printf("%1.8s\n", a);
   printf("%1.9s\n", a);
   printf("%1.10s\n", a);
   printf("%1.11s\n", a);
   printf("%1.12s\n", a);
   return 0;
}
```

➢ Output

```
I
I L
I Lo
I Lov
I Love
I Love N
```

```
I Love Ne
I Love Nep
I Love Nepa
I Love Nepal
```

➢ Discussion and Conclusion

This program successfully displays the desired output using formatted I/O. It initializes a character array a with the given string "I Love Nepal" and then uses printf with a specified width and precision to display the characters incrementally and create a pyramid pattern.

By specifying the width and precision in the format string of printf, we can control the number of characters to be displayed. The width represents the minimum field width, and the precision represents the maximum number of characters to be displayed.