WSD-54

Array:

```c
#include<stdio.h>

int main()
{
int a[10];
int i,j;
for(i=0;i<=10;i++)
{
a[i]=i+2;
}
for(j=0;j<10;j++)
{
printf("values[ %d] of a=%d\n",j,a[j]);

}

return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./array
values[ 0] of a=2
values[ 1] of a=3
values[ 2] of a=4
values[ 3] of a=5
values[ 4] of a=6
values[ 5] of a=7
values[ 6] of a=8
values[ 7] of a=9
values[ 8] of a=10
values[ 9] of a=11
```

String:

```c
#include <stdio.h>

int main ()
{
 char name[10] = {'A', 'm', 'r', 'i', 't','a', '\0'};
 printf(" Message: %s\n", name);
 return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./stringOp
 Message: Amrita
```

```c
#include <stdio.h>
#include <string.h>
int main ()
{
 char name[10] = {'A','m','r','i','t','a','\0'};
 char title[10] = "Mukherjee";
 char total[20];
 int  length ;
 //copy name into total
 strcpy(total, name);
 printf("strcpy( total, name) :  %s\n", total );
 //concatenates name and title
 strcat( name, title);
 printf("strcat( name, title):   %s\n",name );
 // total lenghth of name after concatenation
 length = strlen(name);
 printf("strlen(name) :  %d\n", length );
 return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./stringOp2
strcpy( total, name) :  Amrita
strcat( name, title):   AmritaMukherjee
strlen(name) :  15
```

String compare:

```c
#include<stdio.h>
#include<string.h>
int main()
{
 char strg1[50], strg2[50];
 printf("Enter first string: ");
 scanf("%s",&strg1);
 printf("Enter second string: ");
 scanf("%s",&strg2);
int a;
a=strcmp(strg1, strg2);
 if(a==0)
 {
 printf("\nYou entered the same string two times");
 }
 else
 {
 printf("\nEntered strings are not same!");
 }
 return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ gcc stringOp3.c -o stringOp3
pi@raspberrypi:~/Amrita_C/operator $ ./stringOp3
Enter first string: hi
Enter second string: bye

Entered strings are not same!pi@raspberrypi:~/Amrita_C/operator $ ./stringOp3
Enter first string: same
Enter second string: name

Entered strings are not same!pi@raspberrypi:~/Amrita_C/operator $ ./stringOp3
Enter first string: hi
Enter second string: hi

You entered the same string two timespi@raspberrypi:~/Amrita_C/operator $ gcc stringOp3.c -o stringO
sudo nano stringOp3.c
```

Structure:

```c
#include<stdio.h>
#include<string.h>
struct dimension
{
char name[20];
int length,breadth;
float height;
};
int main()
{
struct dimension fig1,fig2,fig3;
strcpy(fig1.name, "Triangle");  //Specification of fig1
fig1.length=4;
fig1.breadth=3;
fig1.height=2.7;
strcpy(fig2.name, "Square");  //Specification of fig2
fig2.length=5;
fig2.breadth=5;
fig2.height=5.9;
strcpy(fig3.name, "Rectangle"); //Specification of fig1
fig3.length=5;
fig3.breadth=9;
fig3.height=3.2;
printf("Fig1 name: %s\n",fig1.name);  //Printing the details of fig1
printf("Fig1 length: %d\n",fig1.length);
printf("Fig1 breadth: %d\n",fig1.breadth);
printf("Fig1 height: %f\n",fig1.height);
printf("Fig2 name: %s\n",fig2.name);  //Printing the details of fig1
printf("Fig2 length: %d\n",fig2.length);
printf("Fig2 breadth: %d\n",fig2.breadth);
printf("Fig2 height: %f\n",fig2.height);
printf("Fig3 name: %s\n",fig3.name);  //Printing the details of fig1
printf("Fig3 length: %d\n",fig3.length);
printf("Fig3 breadth: %d\n",fig3.breadth);
printf("Fig3 height: %f\n",fig3.height);
return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./structureOp1
Fig1 name: Triangle
Fig1 length: 4
Fig1 breadth: 3
Fig1 height: 2.700000
Fig2 name: Square
Fig2 length: 5
Fig2 breadth: 5
Fig2 height: 5.900000
Fig3 name: Rectangle
Fig3 length: 5
Fig3 breadth: 9
Fig3 height: 3.200000
```

Structures as Function Arguments

```c
#include<stdio.h>
#include<string.h>
struct dimension
{
char name[20];
int length,breadth;
float height;
};
void dim(struct dimension fig); //Globally declare a function
int main()
{
struct dimension fig1,fig2,fig3;
strcpy(fig1.name, "Triangle");  //Specification of fig1
fig1.length=4;
fig1.breadth=3;
fig1.height=2.7;
strcpy(fig2.name, "Square");  //Specification of fig2
fig2.length=5;
fig2.breadth=5;
fig2.height=5.9;
strcpy(fig3.name, "Rectangle"); //Specification of fig1
fig3.length=5;
fig3.breadth=9;
fig3.height=3.2;
dim(fig1);  //Printing the details of fig1
dim(fig2);  //Printing the details of fig2
dim(fig3);  //Printing the details of fig3
return 0;
}
void dim(struct dimension fig)
{
printf("Fig name: %s\n",fig.name);
printf("Fig length: %d\n",fig.length);
printf("Fig breadth: %d\n",fig.breadth);
printf("Fig height: %f\n",fig.height);

}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./structureOp2
Fig name: Triangle
Fig length: 4
Fig breadth: 3
Fig height: 2.700000
Fig name: Square
Fig length: 5
Fig breadth: 5
Fig height: 5.900000
Fig name: Rectangle
Fig length: 5
Fig breadth: 9
Fig height: 3.200000
```

Adding of two array:

```c
#include <stdio.h>
int main()
{
 int a[2][2], b[2][2], add[2][2];
 //declaration of 1st array
printf("Elements of first matrix : \n");
 for (int i = 0; i < 2; i++)
 for (int j = 0; j < 2; j++)
 {
 scanf("%d", &a[i][j]);
 }
//declaration of 2nd array
printf("Elements of second matrix : \n");
 for (int i = 0; i < 2; i++)
 for (int j = 0; j < 2; j++)
 {
 scanf("%d", &b[i][j]);
 }
 for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
 {
 add[i][j] = a[i][j] + b[i][j];
 printf("\nSum Of Matrix:");
 printf("%.d\n", add[i][j]);
}
 printf("\n");
 }
 return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./multi
Elements of first matrix :
1 2 3 4
Elements of second matrix :
1 2 3 4

Sum Of Matrix:2

Sum Of Matrix:4


Sum Of Matrix:6

Sum Of Matrix:8
```

Pointers to Structures:

```c
#include<stdio.h>
#include<string.h>
struct dimension
{
char name[20];
int length,breadth;
float height;
};
void dim(struct dimension *fig); //Globally declare a function
int main()
{
struct dimension fig1,fig2,fig3;
strcpy(fig1.name, "Triangle");  //Specification of fig1
fig1.length=4;
fig1.breadth=3;
fig1.height=2.7;
strcpy(fig2.name, "Square");  //Specification of fig2
fig2.length=5;
fig2.breadth=5;
fig2.height=5.9;
strcpy(fig3.name, "Rectangle"); //Specification of fig1
fig3.length=5;
fig3.breadth=9;
fig3.height=3.2;
dim(&fig1);  //Printing the details of fig1
dim(&fig2);  //Printing the details of fig2
dim(&fig3);  //Printing the details of fig3
return 0;
}
void dim(struct dimension *fig)
{
printf("Fig name: %s\n",fig->name);
printf("Fig length: %d\n",fig->length);
printf("Fig breadth: %d\n",fig->breadth);
printf("Fig height: %f\n",fig->height);

}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./structureOp3
Fig name: Triangle
Fig length: 4
Fig breadth: 3
Fig height: 2.700000
Fig name: Square
Fig length: 5
Fig breadth: 5
Fig height: 5.900000
Fig name: Rectangle
Fig length: 5
Fig breadth: 9
Fig height: 3.200000
```

Union: Accessing Union Members

```c
#include<stdio.h>
#include<string.h>
union dimension
{
char name[20];
int length,breadth;
float height;
};
int main()
{
union dimension fig1;
strcpy(fig1.name,"Triangle");
printf("Fig1 name:%s\n",fig1.name);
fig1.length=4;
printf("Fig1 length:%d\n",fig1.length);

fig1.breadth=3;
printf("Fig1 breadth:%d\n",fig1.breadth);

fig1.height=2.7;
printf("Fig1 height:%f\n",fig1.height);

strcpy(fig1.name,"Triangle");
fig1.length=4;
fig1.breadth=3;
fig1.height=2.7;

printf("Fig1 name:%s\n",fig1.name);
printf("Fig1 length:%d\n",fig1.length);
printf("Fig1 breadth:%d\n",fig1.breadth);
printf("Fig1 height:%f\n",fig1.height);
return 0;
}
```

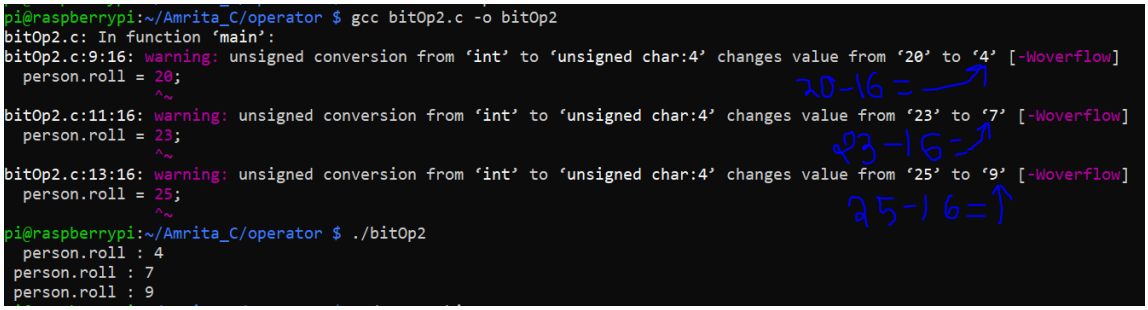
One at a time

All at a time

```
pi@raspberrypi:~/Amrita_C/operator $ ./unionOp1
Fig1 name:Triangle
Fig1 length:4
Fig1 breadth:3
Fig1 height:2.700000
Fig1 name:□□,@ngle
Fig1 length:1076677837
Fig1 breadth:1076677837
Fig1 height:2.700000
```

Bitfields:

```c
#include <stdio.h>
#include <string.h>
struct
{
unsigned int roll : 4;
} person;
int main( )
 {
 person.roll = 20;
 printf( "  person.roll : %d\n",  person.roll );
 person.roll = 23;
 printf( " person.roll : %d\n",  person.roll );
 person.roll = 25;
 printf( " person.roll : %d\n",  person.roll);
 return 0;
}
```




```c
#include <stdio.h>
#include <string.h>
struct
{
unsigned int roll : 3;
} person;
int main( )
 {
 person.roll = 8;
 printf( "  person.roll : %d\n",  person.roll );
 person.roll = 5;
 printf( " person.roll : %d\n",  person.roll );
 person.roll = 9;
 printf( " person.roll : %d\n",  person.roll);
 return 0;
}
```

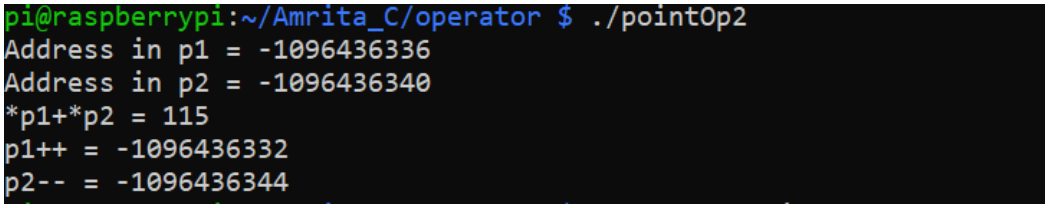
Pointer

```c
#include <stdio.h>
int main()
{
 int *x,* y, a=2,b=5,t;
 printf("Before Swapping\na = %d\nb = %d\n", a, b);
 printf("Swaping of two numbers using pointer:");
 x = &a;
 y = &b;
 t = *x;
 *x = *y;
 *y = t;
 printf("After Swapping\na = %d\nb = %d\n", a, b);
 return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./pointOp1
Before Swapping
a = 2
b = 5
Swaping of two numbers using pointer:After Swapping
a = 5
b = 2
pi@raspberrypi:~/Amrita_C/operator $ sudo nano pointOp1.c
pi@raspberrypi:~/Amrita_C/operator $
```

Arithmatic pointer:

```c
#include <stdio.h>
int main()
{
 int m =15, n = 100, sum, *p1, *p2;
 p1 = &m;             //store the address of m in p1
 p2 = &n;             //store the address of p2
 printf("Address in p1 = %d\n", p1);
 printf("Address in p2 = %d\n", p2);
 sum = *p1+*p2;
 printf("*p1+*p2 = %d\n", sum);
 p1++;
 printf("p1++ = %d\n", p1); //increment pointer
 p2--;
 printf("p2-- = %d\n", p2); //decrement pointer
 return 0;
}
```



Here address has negative values.