

Array in C

1. Write a C program to read and print elements of array – using recursion.

Input:

```
#include <stdio.h>

void readArray(int a[], int i, int n) {
    if (i < n) {
        printf("Enter element %d: ", i + 1);
        scanf("%d", &a[i]);
        readArray(a, i + 1, n);
    }
}

void printArray(int a[], int i, int n) {
    if (i < n) {
        printf("%d ", a[i]);
        printArray(a, i + 1, n);
    }
}

int main() {
    int a[50], n;
    printf("Enter size of array: ");
    scanf("%d", &n);

    readArray(a, 0, n);
    printf("Array elements are: ");
    printArray(a, 0, n);

    return 0;
}
```

Output:

```
Enter size of array: 5
Enter element 1: 1
Enter element 2: -2
Enter element 3: 3
Enter element 4: -4
Enter element 5: 5
Array elements are: 1 -2 3 -4 5
Process returned 0 (0x0)   execution time : 20.352 s
Press any key to continue.
```

2. Write a C program to print all negative elements in an array.

Input:

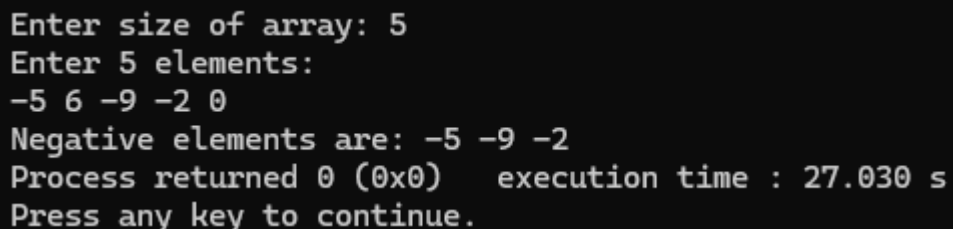
```
#include <stdio.h>

int main() {
    int a[50], n, i;
    printf("Enter size of array: ");
    scanf("%d", &n);

    printf("Enter %d elements:\n", n);
    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);

    printf("Negative elements are: ");
    for (i = 0; i < n; i++) {
        if (a[i] < 0)
            printf("%d ", a[i]);
    }
    return 0;
}
```

Output:



```
Enter size of array: 5
Enter 5 elements:
-5 6 -9 -2 0
Negative elements are: -5 -9 -2
Process returned 0 (0x0)   execution time : 27.030 s
Press any key to continue.
```

3. Write a C program to find sum of all array elements – using recursion.

Input:

```
#include <stdio.h>

int sumArray(int a[], int i, int n) {
    if (i < n)
        return a[i] + sumArray(a, i + 1, n);
    else
        return 0;
}
```

```

int main() {
    int a[50], n, i;
    printf("Enter size of array: ");
    scanf("%d", &n);

    printf("Enter %d elements:\n", n);
    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);

    int sum = sumArray(a, 0, n);
    printf("Sum of all elements = %d", sum);

    return 0;
}

```

Output:

```

Enter size of array: 5
Enter 5 elements:
1 2 3 4 5
Sum of all elements = 15
Process returned 0 (0x0)    execution time : 21.800 s
Press any key to continue.

```

4. Write a C program to find maximum and minimum element in an array – using recursion.

Input:

```

#include <stdio.h>

int findMax(int a[], int i, int n) {
    if (i == n - 1)
        return a[i];
    int max = findMax(a, i + 1, n);
    return (a[i] > max) ? a[i] : max;
}

int findMin(int a[], int i, int n) {
    if (i == n - 1)
        return a[i];
    int min = findMin(a, i + 1, n);
    return (a[i] < min) ? a[i] : min;
}

int main() {

```

```

    int a[50], n, i;
    printf("Enter size of array: ");
    scanf("%d", &n);

    printf("Enter %d elements:\n", n);
    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);

    printf("Maximum = %d\n", findMax(a, 0, n));
    printf("Minimum = %d", findMin(a, 0, n));

    return 0;
}

```

Output:

```

Enter size of array: 5
Enter 5 elements:
3 5 7 9 11
Maximum = 11
Minimum = 3
Process returned 0 (0x0)   execution time : 21.425 s
Press any key to continue.

```

5. Write a C program to search an element in an array using linear search.**Input:**

```

#include <stdio.h>

int main() {
    int a[10] = {10, 2, -1, 0, 3, 7, -5, 9, 4, 6};
    int search_value, i, flag = 0;

    printf("Enter search value: ");
    scanf("%d", &search_value);

    for(i = 0; i < 10; i++) {
        if(search_value == a[i]) {
            flag = 1; // found
            break;
        }
    }

    if(flag == 1)
        printf("Value is found at index %d\n", i);
    else

```

```
        printf("Value not found\n");  
    return 0;  
}
```

Output:

```
Enter search value: 5  
Value not found  
  
Process returned 0 (0x0)   execution time : 6.468 s  
Press any key to continue.
```

Output:

```
Enter search value: 2  
Value is found at index 1  
  
Process returned 0 (0x0)   execution time : 8.762 s  
Press any key to continue.
```

Output:

```
Enter search value: -1  
Value is found at index 2  
  
Process returned 0 (0x0)   execution time : 9.255 s  
Press any key to continue.
```