----------Programs----------

1

```
#include <stdio.h>
void swap(int **a, int **b) {
   int *temp = *a;
   *a = *b;
   *b = temp;
}

int main() {
   int x = 10, y = 20;
   int *ptr1 = &x;
   int *ptr2 = &y;
   printf("Before swap: ptr1 points to %d, ptr2 points to %d\n", *ptr1, *ptr2);
   swap(&ptr1, &ptr2);
   printf("After swap: ptr1 points to %d, ptr2 points to %d\n", *ptr1, *ptr2);

   return 0;
}
```

2

```
#include<stdio.h>
#include<stdlib.h>
void allocateMemory(int **arr,int size){
   *arr=(int*)malloc(size*sizeof(int));
   if(*arr==NULL){
      printf("Memeory not allocated!");
   }
```

```
    for(int i=0;i<size;i++){
        *(arr)[i]=0;
    }
}
int main(){
    int *arr=NULL;
    int n;
    printf("Enter the size of array :");
    scanf("%d",&n);
    allocateMemory(&arr,n);
    if(arr!=NULL){
        printf("the elements are :");
        for(int i=0;i<n;i++){
            printf("arr[%d]\n",*(arr+i));
        }
        free(arr);
        arr=NULL;
    }
    return 0;
}
```

3

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void modify(char **str){
    *str=(char*)malloc(50*sizeof(char));
    if(*str==NULL){
        printf("No memeory allocated !");
```

```
    }
    strcpy(*str,"This is a new string1");
}
int main(){
    char string[10]="hello";
    char *str=&string;
    printf("the orginal string before modifying is :%s\n",str);
    modify(&str);
    if(str!=NULL){
        printf("the string after modifying is :%s",str);
        free(str);
        str=NULL;
    }
    return 0;

}
```

4

```
#include <stdio.h>
void modifyValue(int **ptr) {
    **ptr = 50;
}

int main() {
    int num = 10;
    int *ptr = &num;
    int **doublePtr = &ptr;

    printf("Before modification:\n");
```

```
    printf("Value of num: %d\n", num);
    printf("Value of *ptr: %d\n", *ptr);
    printf("Value of **doublePtr: %d\n", **doublePtr);
    modifyValue(doublePtr);

    printf("\nAfter modification:\n");
    printf("Value of num: %d\n", num);
    printf("Value of *ptr: %d\n", *ptr);
    printf("Value of **doublePtr: %d\n", **doublePtr);

    return 0;
}
```

5

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

int** create2DArray(int rows, int cols) {
    int **arr = (int **)malloc(rows * sizeof(int *));
    if (arr == NULL) {
        printf("Memory allocation failed for rows!\n");
        return NULL;
    }
    for (int i = 0; i < rows; i++) {
        arr[i] = (int *)malloc(cols * sizeof(int));
        if (arr[i] == NULL) {
            printf("Memory allocation failed for row %d!\n", i);
            return NULL;
        }
```

```
    }

    return arr;
}

void free2DArray(int **arr, int rows) {

    for (int i = 0; i < rows; i++) {
        free(arr[i]);
    }
    free(arr);
}

int main() {
    int rows, cols;
    printf("Enter number of rows: ");
    scanf("%d", &rows);
    printf("Enter number of columns: ");
    scanf("%d", &cols);
    int **arr = create2DArray(rows, cols);
    if (arr == NULL) {
        return -1;
    }
    printf("Enter the elements of the 2D array:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("arr[%d][%d]: ", i, j);
            scanf("%d", &arr[i][j]);
        }
```

```
    }
    printf("The 2D array is:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
    free2DArray(arr, rows);

    return 0;
}
```

6

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

int** create2DArray(int rows, int cols) {
    int **arr = (int **)malloc(rows * sizeof(int *));
    if (arr == NULL) {
        printf("Memory allocation failed for rows!\n");
        return NULL;
    }
    for (int i = 0; i < rows; i++) {
        arr[i] = (int *)malloc(cols * sizeof(int));
        if (arr[i] == NULL) {
            printf("Memory allocation failed for row %d!\n", i);
            return NULL;
        }
```

```
    }

    return arr;
}

void free2DArray(int **arr, int rows) {

    for (int i = 0; i < rows; i++) {
        free(arr[i]);
    }
    free(arr);
}

int main() {
    int rows, cols;
    printf("Enter number of rows: ");
    scanf("%d", &rows);
    printf("Enter number of columns: ");
    scanf("%d", &cols);
    int **arr = create2DArray(rows, cols);
    if (arr == NULL) {
        return -1;
    }
    printf("Enter the elements of the 2D array:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("arr[%d][%d]: ", i, j);
            scanf("%d", &arr[i][j]);
        }
```

```
    }
    printf("The 2D array is:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
    free2DArray(arr, rows);

    return 0;
}

7
#include <stdio.h>
#include <stdlib.h>

void setPointer(int **ptr) {
    *ptr = (int *)malloc(sizeof(int));
    if (*ptr == NULL) {
        printf("Memory allocation failed!\n");
        return;
    }

    **ptr = 42;
}

int main() {
    int *ptr = NULL;
```

```
    setPointer(&ptr);
    if (ptr != NULL) {
        printf("Value at the allocated memory: %d\n", *ptr);
        free(ptr);
    }

    return 0;
}

8
#include <stdio.h>
#include <stdlib.h>

void allocateStringArray(char ***arr, int n) {
    *arr = (char **)malloc(n * sizeof(char *));
    for (int i = 0; i < n; i++) {
        (*arr)[i] = (char *)malloc(100 * sizeof(char));
    }
}

int main() {
    int n;
    char **arr;
    printf("Enter the number of strings: ");
    scanf("%d", &n);
    allocateStringArray(&arr, n);
    printf("Enter %d strings:\n", n);
    for (int i = 0; i < n; i++) {
        printf("String %d: ", i + 1);
```

```
        scanf("%s", arr[i]);
    }
    printf("The strings entered are:\n");
    for (int i = 0; i < n; i++) {
        printf("String %d: %s\n", i + 1, arr[i]);
    }
    for (int i = 0; i < n; i++) {
        free(arr[i]);
    }
    free(arr);
    return 0;
}
```

9

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

void modifyStringArray(char **arr, int n) {
    for (int i = 0; i < n; i++) {
        strcat(arr[i], "_modified");
    }
}

int main() {
    int n;
    printf("Enter the number of strings: ");
    scanf("%d", &n);
```

```
    char **arr = (char **)malloc(n * sizeof(char *));
    for (int i = 0; i < n; i++) {
        arr[i] = (char *)malloc(100 * sizeof(char));
    }

    printf("Enter %d strings:\n", n);
    for (int i = 0; i < n; i++) {
        printf("String %d: ", i + 1);
        scanf("%s", arr[i]);
    }

    modifyStringArray(arr, n);

    printf("The modified strings are:\n");
    for (int i = 0; i < n; i++) {
        printf("String %d: %s\n", i + 1, arr[i]);
    }

    for (int i = 0; i < n; i++) {
        free(arr[i]);
    }
    free(arr);

    return 0;
}
```

1

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

```
void reverseString(char *str){
   int start=0;
   int end=strlen(str)-1;
   while(start<end){
      int temp;
      temp=*(str+start);
      *(str+start)=*(str+end);
      *(str+end)=temp;
      start++;
      end--;
   }
}
int main(){
   char string[]="hello";
   reverseString(string);
   printf("the string is %s",string);
}
```

2

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

void concatenateStrings(char *str1, const char *str2) {
   while (*str1) {
      str1++;
   }
   while (*str2) {
      *str1 = *str2;
      str1++;
```

```
        str2++;
    }
    *str1 = '\0';
}

int main() {
    char str1[100] = "Hello";
    char str2[] = "World";

    concatenateStrings(str1, str2);

    printf("Concatenated string: %s\n", str1);

    return 0;
}

3
#include<stdio.h>
#include<string.h>
int calculatelength(char *str){
    int count=0;
    while(*str){
        count+=1;
        str++;
    }
    return count;
}
int main(){
    char string1[]="hello";
```

```
    int result=calculatelength(string1);
    printf("the length is %d",result);
}

4
#include <stdio.h>

int compareStrings(const char *str1, const char *str2) {
    while (*str1 && (*str1 == *str2)) {
        str1++;
        str2++;
    }
    return (unsigned char)(*str1) - (unsigned char)(*str2);
}

int main() {
    char str1[] = "hello";
    char str2[] = "hello";

    int result = compareStrings(str1, str2);
    if (result == 0) {
        printf("The strings are equal.\n");
    } else if (result > 0) {
        printf("str1 is greater.\n");
    } else {
        printf("str2 is greater.\n");
    }

    return 0;
```

```
}
```

5

```
#include <stdio.h>
char* findSubstring(const char *str, const char *sub) {
    if (*sub == '\0') {
        return (char *)str;
    }
    while (*str) {
        const char *s = str;
        const char *p = sub;
        while (*s && *p && (*s == *p)) {
            s++;
            p++;
        }
        if (*p == '\0') {
            return (char *)str;
        }
        str++;
    }
    return NULL;
}
int main() {
    const char *str = "hello world";
    const char *sub = "world";

    char *result = findSubstring(str, sub);
    if (result) {
        printf("Substring found at position: %ld\n", result - str);  // Print the index of the match
```

```
    } else {
        printf("Substring not found.\n");
    }

    return 0;
}
```

6

```
#include <stdio.h>

void replaceChar(char *str, char oldChar, char newChar) {
    while (*str) {
        if (*str == oldChar) {
            *str = newChar;
        }
        str++;
    }
}

int main() {
    char str[] = "hello world";
    replaceChar(str, 'o', 'x');

    printf("Modified string: %s\n", str);

    return 0;
}
```

7

```
#include <stdio.h>

void copyString(char *dest, const char *src) {
    while (*src) {
        *dest = *src;
        dest++;
        src++;
    }
    *dest = '\0';
}

int main() {
    const char *src = "Hello, World!";
    char dest[50];

    copyString(dest, src);

    printf("Source string: %s\n", src);
    printf("Copied string: %s\n", dest);

    return 0;
}
```

8

```
#include <stdio.h>

int countVowels(const char *str) {
    int count = 0;
    while (*str) {
```

```
        if (*str == 'a' || *str == 'e' || *str == 'i' || *str == 'o' || *str == 'u' ||
            *str == 'A' || *str == 'E' || *str == 'I' || *str == 'O' || *str == 'U') {
            count++;
        }
        str++;
    }
    return count;
}

int main() {
    const char *str = "Hello World!";
    int vowelCount = countVowels(str);

    printf("Number of vowels in '%s': %d\n", str, vowelCount);

    return 0;
}
```

9

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

int isPalindrome(const char *str) {
    int start = 0;
    int end = strlen(str) - 1;

    while (start < end) {
        if (str[start] != str[end]) {
            return 0;
```

```
        }
        start++;
        end--;
    }
    return 1;
}

int main() {
    const char *str = "madam";
    if (isPalindrome(str)) {
        printf("'%s' is a palindrome.\n", str);
    } else {
        printf("'%s' is not a palindrome.\n", str);
    }

    return 0;
}

10
#include <stdio.h>
#include <string.h>

void tokenizeString(char *str, const char *delim, void (*processToken)(const char *)) {
    char *token = strtok(str, delim);

    while (token != NULL) {
        processToken(token);
        token = strtok(NULL, delim);
    }
```

```
}

void printToken(const char *token) {
    printf("Token: %s\n", token);

int main() {
    char str[] = "This is a test string!";
    const char *delim = " ";

    tokenizeString(str, delim, printToken);

    return 0;
}

11
#include<stdio.h>
#include<stdlib.h>
int main(){
    int n;
    printf("enter the size of array :");
    scanf("%d",&n);
    int *arr=(int*)malloc(n*sizeof(int));
    for(int i=0;i<n;i++){
        printf("Enter the array element :");
        scanf("%d",arr+i);
    }
    for(int i=0;i<n;i++){
        printf("the element of array =%d\n",arr[i]);
    }
```

```
    free(arr);
    arr=NULL;
    return 0;
}
```

12

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main(){
    char *s=(char*)malloc(100*sizeof(char));
    if(s==NULL){
        printf("Memeory allocation falied");
        return 1;
    }
    printf("enter a string :");
    fgets(s,100,stdin);
    printf("the string that you entered is :%s",s);
    free(s);
    s=NULL;
    return 0;
}
```

13

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

int main() {
    int n;
```

```
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    int *arr = (int*)malloc(n * sizeof(int));
    if (arr == NULL) {
        printf("Memory allocation failed!\n");
        return 1;
    }
    printf("Enter %d integers:\n", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    int *newArr = (int*)realloc(arr, 2 * n * sizeof(int));
    if (newArr == NULL) {
        printf("Memory reallocation failed!\n");
        free(arr);
        return 1;
    }
    for (int i = n; i < 2 * n; i++) {
        newArr[i] = i + 1;
    }
    printf("The array after resizing and filling new elements:\n");
    for (int i = 0; i < 2 * n; i++) {
        printf("%d ", newArr[i]);
    }
    printf("\n");
    free(newArr);

    return 0;
}
```

14

```
#include<stdio.h>
#include<stdlib.h>
void allocateMatrix(int m,int n){
   int **matrix=(int**)malloc(m*sizeof(int *));
   if (matrix == NULL) {
      printf("Memory allocation failed!\n");
      return;
   }
   for (int i = 0; i < m; i++) {
      matrix[i] = (int *)malloc(n * sizeof(int));
      if (matrix[i] == NULL) {
         printf("Memory allocation failed!\n");
         return;
      }
   }
   printf("Enter the elements of the matrix (%dx%d):\n", m, n);
   for (int i = 0; i < m; i++) {
      for (int j = 0; j < n; j++) {
         printf("Enter element at [%d][%d]: ", i, j);
         scanf("%d", &matrix[i][j]);
      }
   }
   printf("The matrix is:\n");
   for (int i = 0; i < m; i++) {
      for (int j = 0; j < n; j++) {
         printf("%d ", matrix[i][j]);
      }
```

```
        printf("\n");
    }
    for (int i = 0; i < m; i++) {
        free(matrix[i]);
    }
    free(matrix);
}
int main(){
    int m, n;
    printf("Enter the number of rows (m): ");
    scanf("%d", &m);
    printf("Enter the number of columns (n): ");
    scanf("%d", &n);

    allocateMatrix(m, n);

    return 0;
}
```

15

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
char* concatStrings(char *string1,char *string2){
    int len1 = strlen(string1);
    int len2 = strlen(string2);
    char *final=(char*)malloc((len1+len2+1)*sizeof(char));
    if(final==NULL){
        printf("memory allocation falied!");
```

```
    }
    strcpy(final,string1);
    strcat(final,string2);
    return final;
}
int main(){
    char s1[10],s2[10];
    printf("Enter the string1:");
    fgets(s1,sizeof(s1),stdin);
    s1[strcspn(s1, "\n")] = '\0';
    printf("Enter the string2:");
    fgets(s2,sizeof(s2),stdin);
    s2[strcspn(s2, "\n")] = '\0';
    concatStrings(s1,s2);
    char *result=concatStrings(s1,s2);
    printf("The final string is :%s",result);
    return 0;
}
```

16

```
#include<stdio.h>
#include<stdlib.h>
int main(){
    int n;
    printf("Enter the size :");
    scanf("%d",&n);
    int *arr=(int*)malloc(n*sizeof(int));
    for(int i=0;i<n;i++){
        printf("Enter the element %d",i+1);
```

```
        scanf("%d",arr+i);
    }
    printf("The array elements are :\n");
    for(int i=0;i<n;i++){
        printf("The array element arr[%d] \n:",arr[i]);
    }
    free(arr);
    arr=NULL;
    return 0;
}
```