Siruri de caractere

1. **Dimensiune unui tablou de siruri de caractere**

#include <stdio.h>

#include <string.h>

int main()

{

char nume[4][10] = {"Miruna","Claudiu","Ionut","Paul"};

int i;

for(i=0;i<4;i++)

printf("%s \n",nume[i]);

printf("\n");

nume[3][0] = 'R'; //modifica primul caracter din al 4-lea cavant

for(i=0;i<4;i++)

printf("%s \n",nume[i]);

printf("\ndimensiunea lui nume este: %lu \n",sizeof(nume)); //40

return 0;

}

**Ex. legat de anteriorul**

#include <stdio.h>

#include <string.h>

int main()

{

char\* nume[4] = {"Miruna","Claudiu","Ionut","Paul"};

int i;

for(i=0;i<4;i++)

printf("%s \n",nume[i]);

printf("\n");

//nume[3][0] = 'R'; NU AM VOIE SA FAC ASTA

printf("\n dimensiunea lui nume este: %lu \n", sizeof(nume)); //32

return 0;

}

**Si inca un ex legat de anteriorul**

#include <stdio.h>

#include <string.h>

int main()

{

char\* nume[4] = {"Miruna","Claudiu","Ionut","Paul"};

char \*\*t = nume;

int i;

for(i=0;i<4;i++)

printf("%s \n",t[i]);

printf("\n");

printf("\ndimensiunea lui t este: %lu \n",sizeof(t)); //8

return 0;

}

Functii pentru prelucrarea șirurilor de aractere

1. Să se scrie o funcție pentru a compara două stringuri, caracter cu caracter. Funcția returnează diferența dintre codurile ASCII ale primelor două caractere diferite, sau 0, dacă cele două șiruri sunt identice.

#include <stdio.h>

#include <string.h>

int comparaSiruri1(char\* s1, char \*s2)

{

int j;

for(j=0; s1[j]==s2[j]; j++)

if(s1[j]=='\0')

return 0;

return s1[j]-s2[j];

}

int comparaSiruri2(char\* s1, char \*s2)

{

for(; \*s1==\*s2; s1++,s2++)

if(\*s1=='\0')

return 0;

return \*s1-\*s2;

}

int main()

{

char s[1000], t[1000];

strcpy(s,"Azi e miercuri");

strcpy(t,"Azi e joi");

int cmp = comparaSiruri1(s,t);

printf("%d %d \n",comparaSiruri1(s,t),comparaSiruri2(s,t*)); //3 3 (j,k,l,m)*

return 0;

}

1. Să se scrie o funcție care să realizeze copierea caracter cu caracter a unui sir in alt sir.

#include <stdio.h>

char\* copiazaSir1(char \*d, char\* s) //d=destinatie, s=sursa

{

int i=0;

while (d[i] = s[i])

i++;

return d;

}

void copiazaSir2(char \*d, char\* s)

{

int i;

while (\*d = \*s)

{

d++;

s++;

}

}

int main()

{

char s[1000], t[1000];

copiazaSir1(s,"Azi e miercuri");

copiazaSir1(t,s);

printf("Sirul t este: %s \n",t); *// Sirul t este: Azi e miercuri*

copiazaSir2(t,s);

printf("Sirul t este: %s \n",t); *// Sirul t este: Azi e miercuri*

return 0;

}

1. Exemplificare pentru functiile strcpy si strncpy, strcmp si strncmp

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main()

{

char \*s="Alina"; char t[]="Alina"; char u[30]; strcpy(u,"Alina");

printf("%ld %ld %ld\n",sizeof(s),sizeof(t),sizeof(u)); //8 6 30

printf("%ld %ld %ld\n", strlen(s),strlen(t),strlen(u)); //5 5 5

**char \*ds=(char\*)strdup(s);**

**puts(ds); //Alina**

**ds[3]='\0';**

**puts(ds); //Ali**

**free(ds);**

strcpy(t,"Emil"); puts(t); //Emil

strncpy(u+4,t+1,3); puts(u); //Alinmil

u[7]='\0'; puts(u); //Alinmil

strcat(u,s+2); puts(u); //Alinmilina

strncat(u,s,3); puts(u); //AlinmilinaAli

printf("%d %d\n", strcmp(s,"Ali"),strncmp(s,"Ali",3)); //110(='n') 0

char \*fs=strchr("Liliana",'a'); puts(fs); // ana

return 0;

}

1. Să se scrie o funcție care realizează conversia unui șir de caractere într-o valoare numerică (biblioteca Ctype: isspace, isdigit, isalpha, isupper, islower, tolower, toupper, etc.).

#include<stdio.h>

#include<string.h>

#include<ctype.h>

int conversie(char \*s)

{

int i, numar, semn;

for(i=0; isspace(s[i]); i++);

semn = (s[i]=='-')?-1:1;

if(s[i]=='+' || s[i]=='-')

i++;

for(numar=0;isdigit(s[i]);i++)

numar=10\*numar+(s[i]-'0');

return semn\*numar;

}

int main()

{

char s[20];

strcpy(s, " -123456789");

printf("Sirul s = %s e transformat in valoarea = %d\n", s, conversie(s));

strcpy(s, " +1234567");

printf("Sirul s = %s e transformat in valoarea = %d\n", s, conversie(s));

strcpy(s, "12345");

printf("Sirul s = %s e transformat in valoarea = %d\n", s, conversie(s));

int numar=conversie(s);

printf("%d", numar+2);

return 0;

}

1. Să se scrie o funcție care să realizeze conversia unui număr întreg într-un șir de caractere.

#include<stdio.h>

#include<string.h>

#include<ctype.h>

void inversare(char[]);//prototip. Precede definitia functiei care o apeleaza

void conversieIA(int n, char s[]){

int j, semn;

if((semn=n)<0)

n=-n;

j=0;

do

s[j++]=n%10+'0';

while ((n/=10)>0);

if(semn<0)

s[j++]='-';

s[j]='\0';

inversare(s); // pt ca n se prelucreaza de la ultima cifra spre prima

}

void inversare(char s[]) //in cadrul aceluiasi sir (i<j)

{ int i,j;

char c;

for(i=0,j=strlen(s)-1;i<j;i++,j--)

c=s[i], s[i]=s[j], s[j]=c;

}

int main()

{

char s[20];

int n = -123456789;

conversieIA(n,s);

printf("Sirul s = %s \n", s);

return 0;

}

1. Să se scrie o funcție care să convertească un număr natural din baza 10 în baza 16.

#include<stdio.h>

#include<string.h>

#include<ctype.h>

void inversare(char[]);

void conversieI16A(int n, char s[]){

int j;

j=0;

char baza16[] ="0123456789abcdef";

do { s[j++] = baza16[n%16]; } while ((n/=16)>0);

s[j]='\0';

inversare(s);

}

void inversare(char s[])

{ int i,j;

char c;

for(i=0,j=strlen(s)-1;i<j;i++,j--)

c=s[i], s[i]=s[j], s[j]=c;

}

int main()

{

char s[20];

int n = 123;

conversieI16A(n,s);

printf("Sirul s = %s \n", s); // Sirul s = 7b

n = 12;

conversieI16A(n,s);

printf("Sirul s = %s \n", s); // Sirul s = c

return 0;

}

1. **Exemplu pentru utilizarea functiei memchr** (The C library function **void \*memchr(const void \*str, int c, size\_t n)** searches for the first occurrence of the character **c** (an unsigned char) in the first **n** bytes of the string pointed to, by the argument **str**.)

#include <stdio.h>

#include <string.h>

int main ()

{

char t[] = "abcdefghijklmnopqrstuvwxyz";

char \*p = memchr(t,'m', 25);

puts(p);

const char s[] = "https://fmi.unibuc.ro/";

const char ch = 'a';

char \*rest;

rest = memchr(s, ch, strlen(s));

if(rest==NULL) printf("Sirul %s nu contine caracterul '%c'.\n", s,ch);

else

printf("Sirul de la primul caracter %c este - %s\n", ch, rest);

return 0;

}

1. **Exemplu cu functia memcompare**. int memcmp(const void \*str1, const void \*str2, size\_t n): **str1** is the pointer to a block of memory, **str2** is the pointer to a block of memory, **n** is the number of bytes to be compared.

#include <stdio.h>

#include <string.h>

int main ()

{

char t[] = "Ana are 6 mere!";

char s[] = "Ana are 2 pere!";

int i = memcmp(t,s,6);

printf("%d \n",i); //0

i = memcmp(t,s,strlen(t));

printf("%d \n\n",i); //4(=6-2)

int v[] = {1,2,5,6,7};

int w[] = {1,2,3,7,8};

i = memcmp(v,w,8);

int x=sizeof(v);

printf("%d %d\n",i, x); //0 20

i = memcmp(v,w,sizeof(v));

printf("%d \n",i); //2(=5-3)

return 0;

}

1. **Exemplu de utilizare a functiei memcpy**: **void \*memcpy(void \*dest, const void \* src, size\_t n)**, where **dest** is pointer to the destination array where the content is to be copied, type-casted to a pointer of type void\*, **src** is pointer to the source of data to be copied, type-casted to a pointer of type void\*, and **n** is the number of bytes to be copied.

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

int a[] = {25, -36, 0, 91, 7415};

int \*b= (int\*) malloc(sizeof(a));

memcpy(b,a,sizeof(a));

int i;

for (i=0;i<sizeof(a)/sizeof(int);i++)

printf("%d ",b[i]);

free(b);

printf("\n\n");

float a1[] = {2.0, 3.5, -1.2};

float b1[3];

memcpy(b1,a1,sizeof(a1));

for (i=0;i<sizeof(a1)/sizeof(float);i++)

printf("%.2f ",b1[i]);

printf("\n\n");

char c[50]="Ana are mere";

memcpy(c+8,c,12); puts(c);

return 0;

}

1. **Exemplu memmove**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

char c[50]="Ana are mere";

memmove(c+8,c,12);

puts(c); //Ana are Ana are mere

char t[] = "...";

memmove(t + 20, t + 13, 16);

puts(t); // memmove este foarte foarte folositor

return 0;

}

1. **Alt ex memmove**

#include <stdio.h>

#include <string.h>

void eliminaAparitii(char \*s, char\* t)

{

char \*p = strstr(s,t);

while(p != NULL)

{

memmove(p,p+strlen(t),s + strlen(s)- (p + strlen(t)) + 1);

p = strstr(s,t);

}

}

int main()

{

char s[100],t[100];

strcpy(s,"abbbccca");

strcpy(t,"bc");

eliminaAparitii(s,t);

printf("%s\n",s); //aa

return 0;

}

1. **Adrese de siruri**

#include<stdio.h>

#include<string.h>

int main()

{

char sir6[10] = "raton";

char \*sir7 = "baton"; //nu am voie sa scriu sir7[0] = 'r';

printf("Adresa lui sir6 este %p \n", sir6);

printf("Adresa lui sir7 este %p \n", sir7);

sir7 = sir6;

puts(sir7);

printf("Adresa lui sir6 este %p \n", sir6);

printf("Adresa lui sir7 este %p \n", sir7);

return 0;

}

1. **Alt ex adrese**

#include<stdio.h>

#include<string.h>

int main()

{

char sir6[10] = "raton";

char sir7[10] = "baton";//acum am voie sa fac sir7[0] = 'r';

sir7[0] = 'r';

printf("Adresa lui sir6 este %p \n", sir6);

printf("Adresa lui sir7 este %p \n", sir7);

sir7 = sir6;

puts(sir7);

printf("Adresa lui sir6 este %p \n", sir6);

printf("Adresa lui sir7 este %p \n", sir7);

return 0;

}

1. **Ex cu strcpy si strncpy.**

#include<stdio.h>

#include<string.h>

int main()

{

char s[10] = "exemplu";

char t[10] = "test";

strncpy(s,t,3); //copiaza in s primele 3 caractere din t

printf("%s \n",s); //tesmplu

s[4] = 0; //pe pozitia 4 marcator de sfarsit de sir

printf("%s \n",s); //tesm

char p[100] = "nimic";

strcpy(p,s); //copiaza in p sirul s, deci p contine tesm

p[3] = '\0'; //pozitia a 3-a este marcator de sfarsit de sir

printf("%s \n",p); //tes

return 0;

}

1. **Exercitiu cu functiile strcmp, strncmp, strcpy si strncpy.**

#include<stdio.h>

#include<string.h>

int main()

{

char s1[20] = "Nor";

char \*s2 = "Noiembrie";

char s3[10], s4[10];

int c = strcmp(s1,s2);

printf("c = %d \n", c);

c = strncmp(s1, s2, 2);

printf("c = %d \n", c);

char \*s23 = strcpy(s3,s2);

printf("s3 = %s \ns23 = %s \n",s3,s23);

printf("s23 contine adresa %p, iar s3 incepe la adresa %p \n", s23,s3);

strncpy(s4,s2,4);

printf("s4 = %s \n", s4);

return 0;

}

1. **Diferenta dintre sizeof si strlen**

#include<stdio.h>

#include<string.h>

int main()

{

char sir[100] = "test";

printf("%lu \n", sizeof(sir)); //100

printf("%lu \n", strlen(sir));//4

return 0;

}

1. **Ex de parcurgere**

#include<stdio.h>

#include<string.h>

int main()

{

char sir[100] = "test";

printf("%lu \n", sizeof(sir));

printf("%lu \n", strlen(sir));

int i;

for(i=0;i<strlen(sir);i++)

printf("%s \n",sir+i);

return 0;

}

1. **Functiile strcat si strncat**

#include<stdio.h>

#include<string.h>

int main()

{

char s[100] = "test";

char t[10] = "miercuri";

char\* p = strncat(s,t,2);

printf("%s \n", s); //testmi

printf("%s \n", p); //testmi

strcat(s,t);

printf("%s \n", s); //testmimiercuri

return 0;

}

1. **Copy**

#include<stdio.h>

#include<string.h>

int main()

{

char s[10] = "exemplu";

char t[10] = "test";

strcpy(t,t+1);

printf("%s \n",t); //est

strcpy(s+1,s);

printf("%s \n",s); //eexemmlu ??

return 0;

}

1. **Functiile strstr si strrstr** function **char \*strrchr(const char \*str, int c)** searches for the last occurrence of the character **c** (an unsigned char) in the string pointed to, by the argument **str**.

#include<stdio.h>

#include<string.h>

int main()

{

char s[] = "exemplu";

char litere[] = {'e','f','g'};

char \*p;

int i;

for(i=0;i<3;i++)

if(p=strchr(s,litere[i]))

{

printf("%s \n",strchr(s,litere[i])); //exemplu

printf("%s \n",strrchr(s,litere[i])); //emplu

}

else

printf("litera %c nu se gaseste in sirul %s \n",litere[i],s);

return 0;

}

1. **Sa se scrie de cate ori apare un sir intr-un alt sir (fara spatii)Strstr**

#include<stdio.h>

#include<string.h>

int main()

{

char s[1000], t[1000];

printf("Sirul s este : ");scanf("%s",s);

printf("Sirul t este : ");scanf("%s",t);

int nrAparitii = 0;

char \*p;

p = strstr(s,t);

while(p)

{

nrAparitii++;

p = strstr(p+strlen(t),t);

}

printf("nrAparitii = %d \n",nrAparitii);

return 0;

}

1. **Functia Strtok**

#include <stdio.h>

#include <string.h>

int main () {

char str[] = "- Ana are mere, Bogdan n-are. Studentii au examen!";

char \*p;

char separatori[] = " ,.-!";

p = strtok(str, separatori);

/\* separa sirul in "tokeni" si afiseaza-i pe linii separate. \*/

while (p != NULL) {

printf("%s\n", p);

p = strtok(NULL, separatori);

}

return 0;

}

1. **Cum calculam lungimea unui sir de caractere fara a utiliza functia strlen.**

#include <stdio.h>

#include<string.h>

int lungimeSir0(char \*s)

{

int i;

for(i = 0; s[i]; i++);

return i;

}

int lungimeSir1(char \*s)

{

int lungime = 0, i;

for(i = 0; s[i]; i++)

lungime++;

return lungime;

}

int lungimeSir2(char \*s)

{

char \*p=s;

while (\*p)

p++;

return p-s;

}

int main()

{

char s[1000];

strcpy(s,"Azi e miercuri");

printf("lungime sir = %d \n",lungimeSir0(s));

printf("lungime sir = %d \n",lungimeSir1(s));

printf("lungime sir = %d \n",lungimeSir2(s));

return 0;

}