Laborator 9 – Exerciții rezolvate

Să se rezolve prin realizarea programului C următoarele probleme:

Problema 1. Folosind pointeri să se scrie o funcție C care returnează lungimea unui șir de caractere (funcția strlen –cu pointeri!).

Rezolvare.

Varianta cu pointeri:

Utilizam un pointer auxiliar p ce este inițializat să pointeze pe primul caracter din s. In cadrul buclei while este examinat fiecare caracter până se întâlnește caracterul '\0'care semnifică sfârșitul șirului de caractere iar apoi se scad cele 2 adrese. Este posibila omiterea testului explicit iar astfel de bucle se scrise adesea

```
while (*p) p++;
```

Deoarece p pointează spre caractere, p++ face ca p sa avanseze de fiecare data pe caracterul următor, iar p-s dă numărul de caractere parcurse, adică lungimea șirului.

```
#include <stdio.h>

int mystrlen(char *s)//returnează lungimea sirului s
{    char *p = s;//la declarare, p este initializat să pointeze pe primul
caracter din s
     while (*p != '\0') p++;
     //la ieșirea din bucla, p este poziționat după ultimul caracter al șirului
     //(pe delimitatorul '\0')
     return(p-s);
     //p-s este numărul de elemente dintre p și s
}

int main(void)
{
  char* s = "Ana are trei mere.";
  printf("Sirul '%s' este compus din %d caractere.", s, mystrlen(s));
  return 0;
}
```

Rulare:

```
Șirul 'Ana are trei mere.' este compus din 18 caractere.
```

Varianta cu tablouri:

```c
#include <stdio.h>

int mystrlen(char a[])
{  int n;
   for(n=0;a[n];n++); //ciclul for se oprește în momentul întâlnirii octetului null care termină orice șir de caractere
   return(n);
}

int main(void) {
  char sir[]= "Hello World";
  printf("lungime sirului '%s' este % i", sir, mystrlen(sir));
  return 0;
}
```

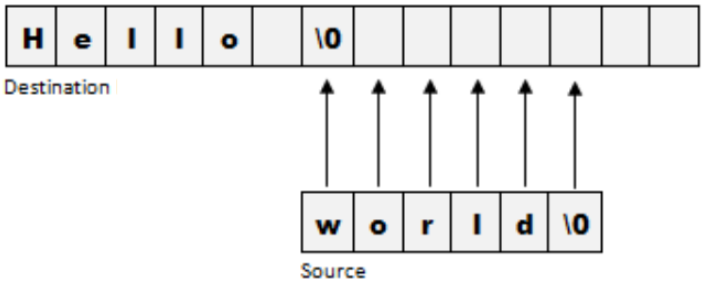
Rulare:

```
Lungime șirului 'Hello World' este  11
```

Problema 2. Folosind pointeri să se scrie o funcție C care primind ca și parametru 2 șiruri de caractere, le concatenează cu obținerea noului sir in primul argument al funcției (implementare pentru funcția strcat – cu pointeri!).

Rezolvare. Funcția mystrcat concatenează șirul sursă la șirul destinație. Funcția returnează șirul destinație. Șirul destinație trebuie să aibă suficientă memorie alocată pentru a acomoda șirul rezultat. Marcatorul de sfârșit de sir ('\0') al șirului destinație este șters si in locul sau e copiat primul caracter din șirul sursa. Vom copia mai apoi toate celelalte caractere din șirul sursa inclusiv marcatorul de sfârșit al acestuia ('\0').

strcat



Varianta cu pointeri:

Metoda 1

```c
#include <stdio.h>

char* mystrcat(char* dest, char* s)
{ // il facem pe p sa puncteze spre sfarsitul lui dest
```

```
  char* p = dest;
  while(*p)p++;//sau while(*p!='\0') p++;
  // Adaugam caracterele lui s la finalul lui dest
  while((*p++=*s++)); // sau while(*s!='\0')*p++=*s++;*p='\0';
  //strcat() intoarce adresa inceputului sirului dest
  return dest;
}

int main ()
{
  char dest[100]= "Hello ";
  mystrcat(dest, "World");
  mystrcat(dest, ". How ");
  mystrcat(dest, "are ");
  printf("%s",mystrcat(dest, "you?"));
  return 0;
}
```

Rulare:

```
Hello world. How are you?
```

Metoda 2

```
char* mystrcat(char *a, char *b)
{ char* s = a;
  for(;*a;a++); //ajungem la sfârșitul șirului a
  for(;*a=*b;a++,b++); //ieșim din acest ciclu for când ajungem la sfârșitul
șirului
  //b adică vom atribui valoarea 0, care constituie terminatorul șirului b
  return s;
}
```

Problema 3. Folosind pointeri să se scrie o funcție C care primind ca si parametru 2 șiruri de caractere returnează un număr negativ dacă primul sir este mai mic, 0 dacă cele 2 șiruri sunt egale si un număr pozitiv dacă primul sir este mai mare decât al 2-lea. Comparațiile sunt lexicografice. (implementare pentru funcția strcmp–cu pointeri!).

Rezolvare. Funcția mystrcmp(s, t) compara șirurile de caractere s și t și returnează un număr negativ, zero sau pozitiv în funcție de relația lexicografică dintre s și t; care poate fi: s<t, s=t sau s>t. Valoarea returnată este obținută prin scăderea caracterelor ASCII de pe prima poziție unde s diferă de t.

Varianta cu pointeri:

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

```c
// returneaza <0 daca s<t,
// 0 daca s==t,
// >0 daca s>t
int mystrcmp(char *s, char *t) {
  //pana cand perechile de caractere curente din s si t sunt egale
  //ne pozitionam pe urmatoarea pereche de caractere
  for (; *s == *t; s++, t++)
    if (!*s)//*s!='\0'
      // daca am ajuns la sfarsitul amebele siruri si ele sunt identice
      // returnam 0
      return(0);
  // returnam diferența dintre codurile ASCII ale primei perechi de caractere
diferite
  return(*s - *t);
}


int main()
{
  int rez;
  char* s = "Ana";
  char* t = "Anamaria";

  rez = mystrcmp(s, t);

  if (rez > 0)
    printf("%s e mai mare decat %s",s,t);
  else if (rez < 0)
    printf("%s e mai mic decat %s",s,t);
  else
    printf("%s e egal cu %s",s,t);

  return 0;
}
```

Rulare:

```
Șirul Ana e mai mic decât șirul Anamaria
```