Laboratorul 9 – Șiruri de caractere

```
/// 3.2. Să se scrie o funcţie care să realizeze extragerea dintr-un şir de caractere
sursă a unui subşir specificat prin poziţia în cadrul sursei şi a numărului de caractere
extrase. Se va scrie şi programul care testează această funcţie.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

void subsir(char *d, const char *s, int poz, int nr){
    if(poz>=strlen(s)) d[0]='\0';
    else {
        strncpy(d, s+poz, nr);
        d[nr]='\0';
    }
}

int main()
{
    char s[N], d[N];
    int poz, nr;

    printf("Sirul din care copiem: ");
    fgets(s, N, stdin);
    printf("Pozitia de unde copiem: ");
    scanf("%d", &poz);
    printf("Numar caractere de copiat: ");
    scanf("%d", &nr);
    subsir(d, s, poz, nr);
    printf("Subsirul extras: %s\n", d);

    return 0;
}
```

```c
/// 3.4. Să se scrie o funcţie pentru ştergerea unui subşir dintr-un şir de caractere dat. Subşirul se va specifica prin poziţie şi număr de caractere. Se va scrie şi programul care testează această funcţie.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

void sterge(char *d, int poz, int nr){
    if(poz>=strlen(d)) return ;
    if(poz+nr>=strlen(d)) d[poz]='\0';
    else memmove(d+poz, d+poz+nr, strlen(d+poz+nr)+1);
}

int main()
{
    char s[N];
    int poz, nr;

    printf("Sirul: ");
    fgets(s, N, stdin);

    printf("Pozitia de unde stergem: ");
    scanf("%d", &poz);

    printf("Numarul de caractere sterse: ");
    scanf("%d", &nr);

    sterge(s, poz, nr);

    printf("Subsirul ramas: %s\n", s);

    return 0;
}
```

```c
/// 3.3. Să se scrie o funcţie pentru inserarea unui şir de caractere sursă într-un şir de caractere destinaţie, specificând poziţia din care începe inserarea. Se va scrie şi programul care testează această funcţie.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

void insert(char *d, const char *s , int poz){
    if(poz>=strlen(d)) strcat(d, s);
    else {
        char aux[N];
        strcpy(aux, d+poz);
        strcpy(d+poz, s);
        strcat(d, aux);
    }
}

int main()
{
    char s[N], d[N];
    int poz, nr;

    printf("Sirul initial: ");
    fgets(s, N, stdin);

    printf("Sirul pe care il introducem: ");
    fgets(d, N, stdin);

    printf("Pozitia unde inseram: ");
    scanf("%d", &poz);

    if(s[strlen(s)-1]=='\n') s[strlen(s)-1]='\0';
    insert(s, d, poz);

    printf("Sirul final: %s\n", d);

    return 0;
}
```

```c
/// 3.5. Să se scrie o funcţie pentru a verifica dacă un şir dat este subşir al unui alt şir de caractere. În caz afirmativ, se va specifica poziţia pe care se regăseşte pentru prima dată. Se va scrie şi programul care testează această funcţie.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

int pozSubsir(const char *s, const char *subsir){
    char *p = strstr(s, subsir);

    if(p!=NULL) return p-s;
    else return -1;
}

int main()
{
    int poz, i=0, nrAparitii=0;
    char s[N], subsir[N];

    printf("Sirul principal: ");
    gets(s);

    printf("Subsirul cautat: ");
    gets(subsir);

    while((poz=pozSubsir(s+i, subsir))>=0){
        nrAparitii++;
        printf("%d ", poz+i);
        i+=poz+1;
    }

    if(nrAparitii==0) printf("Subsirul nu se gaseste in sir");

    return 0;
}
```

```c
/// 3.6. Să se scrie două funcţii, una care converteşte un număr întreg sau real într-un şir de caractere, iar cealaltă face operaţia inversă. Se va scrie şi programul care testează aceste funcţii.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

void inverseaza(char s[]){
    int i, n=strlen(s);
    for(i=0; i<n/2; i++){
        char aux=s[i];
        s[i]=s[n-i-1];
        s[n-i-1]=aux;
    }
}

char *intToStr(int nr){
    char aux[N];
    int i=-1, cifra, semn=1;

    if(nr<0) semn=-1, nr=-nr;
    else if(nr==0) semn=0;

    while(nr>0){
        cifra=nr%10;
        nr/=10;
        aux[++i]=cifra+'0';
    }

    if(semn==-1) aux[++i]='-';
    else if(semn==0) i=0, aux[0]='0';

    aux[++i]='\0';
    inverseaza(aux);

    char *s=calloc(strlen(aux)+1, sizeof(char));
    if(s==NULL){
        printf("Memorie insuficienta");
        exit(1);
    }

    strcpy(s, aux);
    return s;
}

int main(){

    int nr;
    printf("Introduceti numarul: ");
    scanf("%d", &nr);
    char *s=intToStr(nr);

    printf("%s", s);
    free(s);

    return 0;
}
```

```c
/// 3.7. Să se scrie un program care citește n șiruri de caractere și afișează șirul cel mai lung și șirul cel mai mare alfanumeric.

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

char **citeste(int *pn){  /// ** pentru ca este vector de caractere (cum ar fi c[][])
    int i;
    char tmp[100];

    printf("Numar de siruri: ");
    scanf("%d%*c", pn);

    char **s=calloc(*pn, sizeof(char*));
    if(s==NULL){
        printf("Memorie insuficienta");
        exit(1);
    }

    for(i=0; i<*pn; i++){
        fgets(tmp, 100, stdin);
        s[i]=calloc(strlen(tmp)+1, sizeof(char));
        if(s[i]==NULL){
            printf("Memorie insuficienta");
            exit(1);
        }
        strcpy(s[i], tmp);
    }

    return s;
}

void *celMaiLungSir(char **s, int n){
    int nMax=0, i;
    char *cMax=malloc(sizeof(char)*n+1);

    for(i=0; i<n; i++)
        if(strlen(s[i])>nMax){
            nMax=strlen(s[i]);
            strcpy(cMax, s[i]);
        }

    return cMax;
}

void afisare(int n, char **s){
    for(int i=0; i<n; i++) printf("%s", s[i]);
}

int main(){
    int n;
    char **s, *cmls;
    s=citeste(&n);
    cmls=celMaiLungSir(s, n);
    printf("Cel mai lung sir este sirul: %s", cmls);
    free(s);
    return 0;
}
```