Alokace paměti – dynamické proměnné - motivace

- standardní automatické lokální proměnné se vytváří na zásobníku. Jelikož zásobník mívá často omezenou velikost (délku), může (zvláště při použití polí) dojít k jeho přetečení
- proměnné, se kterými se pracuje, mívají různou délku pro různé okamžiky (texty, vyhodnocení dat …)
- z předchozího vyplývá nutnost (vhodnost) práce s daty, které dynamicky mění svou velikost. Především u polí se tento přístup jeví jako nutnost.
- Pro práci potřebujeme mít možnost požádat o paměť (alokace zdroje), a vrátit paměť (dealokace, odalokování zdroje). Každá alokace by měla mít svou odalokaci – v C není automatický mechanizmus, musí zařídit programátor.

Alokace paměti – dynamické proměnné

- je možné alokovat paměť o dané velikosti bytů – která nemusí být konstantní. Velikost se určuje až za chodu programu.
- velikost typu je nutné "zjistit" pomocí klíčového slova sizeof(typ)
- jelikož se alokuje "univerzální" blok paměti, je návratová hodnota typu *void a je nutné ji přetypovat
- paměťový blok je nutné uvolnit
- kontrola mezí není prováděna

```
#include <stdlib.h>

int *pi = NULL;

pi = (int *) malloc (počet * sizeof(int)); // vrátí ukazatel na počátek bloku paměti
// malloc vrací typ void*
if (pi != NULL) free(pi);  // odalokuje paměťový blok daný ukazatelem
```

Napište funkci, která vrátí první větu (kopii) textu pomocí dynamického řetězce. Statické pole nelze vracet, zároveň neznáme délku první věty, proto musíme použít pole alokované.

```
char Text[]="Bylo jaro. Ale venku byla stále zima.";
char *Vysledek ;

Vysledek = VratPrvniVetu(Text);
```

```
// funkce vracející první větu z textu předaného jako parametr
// vrací ukazatel na nově naalokovaný blok paměti
char * VratPrvniVetu(char text[])
{
  int delka=0,i;
 char *vyslret=NULL;

for(delka=0;text[delka]!=´\0´;delka++) // zjištění délky první věty
  if(text[delka]==´.´)
   {
      delka++; break;
   }
vyslret=(char*) malloc((delka+1) * sizeof(char)); // alokace paměti pro kopii textu
if (vyslret == NULL) return NULL;
for(i=0;i<delka;i++)    // kopírování první věty
   *(vyslret+i) = text[i];
*(vyslret+delka) =´\0´;
 return vyslret;
}
```

```
char Text[]="Bylo jaro. Ale venku byla stále zima.";
// délka a ukončovací znak se doplní překladačem
char *Vysledek  = NULL; // proměnná pro uložení výsledku - inicializovat

Vysledek = VratPrvniVetu(Text); // volání funkce
// – parametrem je ukazatel na počátek textu
// - vrací se ukazatel na řetězec, který obsahuje kopii textu první věty

if  (Vysledek) // != NULL => byl naplněn => byl alokován
 {
   free(Vysledek) ; // vrátí se paměť, která již nebude využívána
Vysledek = NULL; // signál pro případné další použití  => nealokováno
}
```

```
// alternativa s předáním návratového řetězce v poli parametrů
//funkce vracející první větu z textu předaného jako parametr, ve druhém parametru
// návratová hodnota 0=v pořádku, jinak chyba
int  VratPrvniVetu(char text[], char **navrat)
{
char *vyslret=NULL;
*navrat = NULL;

vyslret=(char*) malloc((delka+1) * sizeof(char)); // alokace paměti pro kopii textu
if (vyslret == NULL) return 1;

*navrat = vyslret;
return 0;
}

char Text[]="Bylo jaro. Ale venku byla stále zima.";
char *Vysledek  = NULL; // proměnná pro uložení výsledku - inicializovat

int vysl = VratPrvniVetu(Text, &Vysledek); // volání funkce
```