Příkazy skoku:

- **break** - Vyskočí z jednoho cyklu.
- **continue** - Vrátí se na podmínku.
- **goto** -
- **return** - Výhoda oproti break je, že se dá použít ve vícei zanořených cyklech a vyskočí ze všech.

Datový typ ukazatel, alokace paměti:

Alokace paměti:

Virtuální paměťový prostor je 2^{64}. Do UPP se zapíše kód a statické proměnné, které jsou pouze ke čtení, a dále heap a stack (zásobník).

- **Heap** - Dynamická alokace paměti.
- **Stack** - Paměť omezena velikostí, ukládá lokální proměnné.

Příklad alokace paměti:

```
int example = 51;  // Zapisuje se do stacku
```

Ukazatel:

Ukazatel může ukazovat kamkoliv, ale musíme si dát pozor, aby neukazoval na neplatnou paměť.
Příklad práce s ukazatelem:

```
\int *ptr = &example;  // U l o    adresu hodnoty example
\*ptr = 51;            // Dereference, z  s k   hodnotu na adrese
ptr = 0;              // Ukazuje na adresu 0, NULL
```

Bazový typ ukazatele určuje velikost paměti, kterou ukazuje:

- **&** - Vrací adresu proměnné.
- ***** - Dereference, vrací hodnotu na adrese.

Dynamická alokace paměti:

Na heapu je nutné použít ukazatele, protože nelze přímo vytvářet pojmenované proměnné. Programátor je zodpovědný za správnou alokaci i dealokaci paměti.
Příklad dynamické alokace:

```
int \*pi = malloc(sizeof(int));  // Alokuje  p a m    na heapu
if (pi == NULL) {
    return ERR_MALLOC;  // O v     , zda malloc  u s p  l
}
free(pi);  // U v o l n   p a m
```

Musíme dávat pozor na ztrátu ukazatele, jinak vzniká **memory leak**.

Pole:

Pole je datový typ, který ukládá prvky stejného typu. V C není kontrola mezí polí, což může vést k **buffer-overflow** chybám.