BPC-UDP Cvičení 10

# Příklad 1

Základy práce s pointery.

Ve funkci main():

1. Definujte proměnné x a y typu **int** a inicializujte je na hodnoty **10** a **20**.
2. Definujte proměnné ptr\_x a ptr\_y typu pointer na **int** a inicializujte je na hodnotu NULL.
3. Uložte do proměnné ptr\_x adresu proměnné x a do proměnné ptr\_y adresu proměnné y.
4. Vytiskněte hodnoty proměnných x, y, ptr\_x, ptr\_y na standardní výstup.
5. Vytiskněte adresy proměnných x, y, ptr\_x, ptr\_y na standardní výstup.
6. Zakreslete adresy proměnných a jejich hodnoty do paměťové mapy.
7. Pomocí ukazatelů ptr\_x a ptr\_y vyměňte obsah proměnných x a y.
8. Po výměně vytiskněte hodnoty proměnných x a y na standardní výstup.

# Příklad 2

Napište a odlaďte program v jazyce C, který bude používat pro vzájemnou výměnu obsahu dvou proměnných funkci nazvanou swap\_numbers().

1. Napište deklaraci (úplný funkční prototyp) funkce swap\_numbers(). Funkce bude mít dva formální parametry aPtr1, aPtr2, oba typu ukazatel na proměnnou typu **int**. Návratová hodnota funkce bude typu **void**.
2. Napište definici funkce swap\_numbers().
3. Ve funkci main():
   1. Definujte proměnné x a y typu **int** a inicializujte je na hodnoty 10 a 20.
   2. Vytiskněte hodnoty proměnných x a y na standardní výstup.
   3. Zavolejte funkci swap\_numbers(). Jako argumenty ji předejte adresy proměnných x a y.
   4. Po výměně vytiskněte hodnoty proměnných x a y na standardní výstup.

# Příklad 3

Pokračujte ve zdrojovém textu z příkladu 2 následujícími úkoly.

1. Ve funkci main():
   1. Napište definici proměnné ptr\_func typu ukazatel na funkci, která má dva formální parametry aPtr1, aPtr2, oba typu ukazatel na proměnnou typu **int** a návratovou hodnotu typu **void**
   2. Nastavte hodnotu proměnné ptr\_func tak, aby obsahovala adresu funkce swap\_numbers().
   3. Zavolejte funkci swap\_numbers() pomocí ukazatele ptr\_func.
   4. Napište deklaraci (úplný funkční prototyp) funkce print\_numbers(). Funkce bude mít dva formální parametry aPtr1, aPtr2, oba typu ukazatel na proměnnou typu **int**. Návratová hodnota funkce bude typu **void**.
   5. Napište definici funkce print\_numbers(). Funkce vytiskne na standardní výstup hodnoty proměnných typu **int** předaných pomocí aPtr1, aPtr2.
   6. Nastavte hodnotu proměnné ptr\_func tak, aby obsahovala adresu funkce print\_numbers().
   7. Zavolejte funkci print\_numbers() pomocí ukazatele ptr\_func.

# Příklad 4

Napište a odlaďte program v jazyce C, který nalezne největší a nejmenší číslo v poli čísel. (Vyjděte z příkladu 1, cvičení 7.) **V projektu nutno nastavit překladač CLANG**.

1. Napište deklaraci (úplný funkční prototyp) VLA funkce minmax(), která nalezne v poli čísel nejmenší a největší číslo. Funkce bude mít 4 formální parametry:  
   - aNum typu **size\_t** reprezentující počet elementů pole,  
   - aData pole elementů typu **double**,  
   - aPtrMin a aPtrMax ukazatele na proměnné typu **double**.  
   Návratová hodnota funkce bude typu **int**.
2. Napište definici VLA funkce minmax().  
   Bude-li funkci předán některý argument typu ukazatel s hodnotou **NULL** nebo argument vložený do parametru aNum bude **0**, funkci ukončete s návratovou hodnotu **-1** (která signalizuje chybné argumenty).  
   Funkce najde největší a nejmenší číslo v poli aData, nejmenší číslo zapíše do proměnné, na kterou ukazuje pointer aPtrMin, největší číslo zapíše do proměnné, na kterou ukazuje pointer aPtrMax.  
   Pokud funkce proběhla úspěšně, je návratová hodnota funkce **0**.
3. Ve funkci main():
   1. Definujte pole data pro uložení N čísel typu **double**. Hodnotu N nastavte na **10** pomocí direktivy **#define**.
   2. Definujte proměnnou data\_num typu **size\_t**.
   3. Vytiskněte na standardní výstup výzvu uživateli „*Zadej pocet cisel: “*.
   4. Přečtěte ze standardního vstupu číslo typu **size\_t** a uložte ho do proměnné data\_num. Pokud při čtení dojde k chybě, program ukončete s návratovým kódem **1**.
   5. Pokud bude hodnota uložená do data\_num větší než N, program ukončete s návratovým kódem **2**.
   6. Vytiskněte na standardní výstup výzvu uživateli *„Zadej X cisel: “* (za *X* dosaďte skutečnou hodnotu).
   7. Přečtěte ze standardního vstupu data\_num čísel typu **double** a uložte je do pole pojmenovaného data. Pokud při čtení dojde k chybě, program ukončete s návratovým kódem **3**.
   8. Definujte proměnné min a max typu **double**.
   9. Zavolejte funkci minmax(). Jako argumenty ji předejte: data\_num, data a adresy proměnných min a max.  
      Pokud funkce vrátí hodnotu **-1**, program ukončete s návratovým kódem **4**.  
      Jinak vytiskněte na standardní výstup hodnoty proměnných min a max a program ukončete s návratovým kódem **0**.

# Příklad 5

Upravte program z příkladu 4 tak, aby pole pro uložení čísel bylo alokováno dynamicky podle zadaného počtu zpracovávaných čísel.

1. Založte nový projekt a přidejte do něj soubory main.c, check.h a check.c. Soubor main.c převezměte z příkladu 3.
2. Ve funkci main():
   1. Zrušte proměnnou data.  
      Definujte proměnnou ptr\_data typu pointer na **double**Proměnnou data\_num typu **size\_t** ponechte beze změny.
   2. Vytiskněte na standardní výstup výzvu uživateli „*Zadej pocet cisel*: “. Přečtěte ze standardního vstupu číslo typu **size\_t** reprezentující počet zadávaných čísel a uložte ho do proměnné data\_num. Pokud při čtení dojde k chybě, program ukončete s návratovým kódem **1**.
   3. Dynamicky alokujte pole pro uložení zadaného počtu čísel typu **double**. Na začátek pole bude ukazovat pointer ptr\_data. Pokud by alokace selhala, vypište na standardní chybový výstup chybovou hlášku „*Chyba alokace!*“ a program ukončete s návratovým kódem **2**.
   4. Vytiskněte na standardní výstup výzvu uživateli *„Zadej X cisel: “* (za *X* dosaďte skutečnou hodnotu).
   5. Přečtěte ze standardního vstupu data\_num čísel typu **double**. Čísla ukládejte do dynamicky alokovaného pole, na jehož začátek ukazuje ukazatel ptr\_data. Pokud při čtení dojde k chybě, uvolněte dynamicky alokovanou paměť a program ukončete s návratovým kódem **3**.
   6. Zavolejte funkci minmax(). Jako argumenty ji předejte: data\_num, ptr\_data a adresy proměnných min a max.
   7. Pokud funkce minmax() vrátila hodnotu **-1**, uvolněte dynamicky alokovanou paměť a program ukončete s návratovým kódem **4**.
   8. Uvolněte dynamicky alokovanou paměť.
   9. Vytiskněte na standardní výstup hodnoty proměnných min a max a program ukončete s návratovým kódem **0**.

# Příklady pro samostatné procvičení probrané látky

# Příklad 6

Čtení vstupních dat (body b) až e)) z příkladu 5 přesuňte do funkce read\_data().

1. Napište deklaraci (úplný funkční prototyp) funkce read\_data(). Funkce bude mít jeden parametr aPtrNum pointer na typ **size\_t**.  
   Návratová hodnota funkce bude typu pointer na typ **double**.
2. Napište definici funkce read\_data().
   1. Zkontrolujte, zda funkci byly předány korektní hodnoty argumentů:
      1. Bude-li mít pointer aPtrNum, funkci ukončete s návratovou hodnotou **NULL**.
   2. Deklarujte proměnnou data\_num typu **size\_t** a ptr\_data typu pointer na typ **double**.
   3. Vytiskněte na standardní výstup výzvu uživateli „*Zadej pocet cisel*: “. Přečtěte ze standardního vstupu číslo typu **size\_t** reprezentující počet zadávaných čísel a uložte ho do proměnné data\_num.
   4. Dynamicky alokujte pole pro uložení zadaného počtu čísel typu **double**. Na začátek pole bude ukazovat pointer ptr\_data. Pokud by alokace selhala, vypište na standardní výstup chybovou hlášku „*Chyba alokace!*“ a funkci ukončete s návratovým kódem **NULL**.
   5. Přečtěte ze standardního vstupu data\_num čísel typu **double**. Čísla ukládejte do dynamicky alokovaného pole, na jehož začátek ukazuje ukazatel ptr\_data. Pokud při čtení dojde k chybě, uvolněte dynamicky alokovanou paměť a funkci ukončete s návratovým kódem **NULL**
   6. Předejte prostřednictvím pointeru aPtrNum hodnotu proměnné data\_num a prostřednictvím návratové hodnoty předejte hodnotu proměnné ptr\_data.
3. Ve funkci main() zavolejte funkci read\_data(). Pokud funkce vrátí hodnotu **NULL**, program ukončete s návratovým kódem **4**. Jinak vykonejte body f )až h) z příkladu 5.