# Volání Fukcí v C++

**Vypracování je v přísl. Adresáři podle čísla úkolu.**

## 1) a 2) viz *main.cpp*, *main.exe*, *main.obj*, *main.asm*.

## 3)

Assemblerovský kód vygenerovaný kompilátorem Borland (C++ builder 5.0), který odpovídá kódu v ***main.obj***. (celý kód v ***main.asm***)

```
;
   ;  int main(void)
   ;
      push      ebp
      mov       ebp,esp
      add       esp,-8
   ;
   ;  {
   ;
   ;    div_t x;              // div_t je struct
obsahující 2 inty.
   ;    x = div(255,16);      // deleni se zbytkem
   ;
@1:
      push      16
      push      255
      lea       eax,dword ptr [ebp-8]
      push      eax
      call      @_div
      add       esp,12
   ;
   ;    printf("%d:%d",x.quot,x.rem);
   ;
      push      dword ptr [ebp-4]
      push      dword ptr [ebp-8]
      push      offset s@
      call      @_printf
      add       esp,12
   ;
   ;    return 0;
   ;
      xor       eax,eax
   ;
   ;  }
   ;
```

Assemblerovský kód **exe** souboru, který představuje funkci **main**. Zde se volá knihovní funkce **div**. Funkce provede celočíselné dělení prvního argumentu druhým.

Kód je vypsán debugerem C++ builderu, protože najít těchto pár instrukcí v souboru **main.exe.asm** (soubor je vygenerovaný pomocí **objdump.exe -d main.exe > main.exe.asm**) je velmi obtížné.

Je zde krásně vidět, že je použito volání funkce podle konvence jazyka C, kdy se na zásobník nejprve uloží poslední argument (tj. číslo 16 = 0x10), pak předposlední atd. až se nakonec uloží první (tj. 255 = 0xFF). Nakonec se zavolá funkce pomocí instrukce **call.**

Po návratu se musí funkce main postarat o smazání hodnot ze zásobníku.

```
0040114F 90              nop
main.cpp.4: int main(void)
00401150 55              push ebp
00401151 8BEC            mov ebp,esp
00401153 83C4F8          add esp,-0x08
main.cpp.8: x = div(255,16);
00401156 6A10            push 0x10
00401158 68FF000000      push 0x000000ff
0040115D 8D45F8          lea eax,[ebp-0x08]
00401160 50              push eax
00401161 E88C010000      call CC3250._div
00401166 83C40C          add esp,0x0c
main.cpp.9: printf("%d:%d",x.quot,x.rem);
00401169 FF75FC          push dword ptr [ebp-0x04]
0040116C FF75F8          push dword ptr [ebp-0x08]
0040116F 68BC204000      push 0x004020bc
00401174 E885010000      call CC3250._printf
00401179 83C40C          add esp,0x0c
main.cpp.10: return 0;
0040117C 33C0            xor eax,eax
main.cpp.11: }
0040117E 59              pop ecx
0040117F 59              pop ecx
00401180 5D              pop ebp
00401181 C3              ret
```

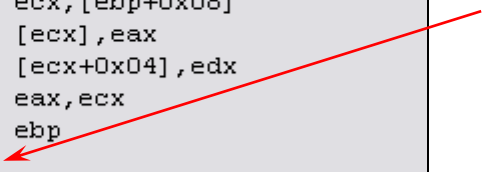
**Poslední argument** a potom **první argument**

**volání funkce div**

**vrácení zásobníku na původní hodnotu zvýšením *stack-pointru***

Pro úplnost ještě asm. kód volané funkce **div**, která zpracuje hodnoty ze zásobníku, ale nechá je tam. Odstranit je musí fce **main**.

```
325611F7 C3              ret
cc3250._div:
325611F8 55              push ebp
325611F9 8BEC            mov ebp,esp
325611FB 8B450C          mov eax,[ebp+0x0c]
325611FE 99              cdq
325611FF F77D10          idiv dword ptr [ebp+0x10]
32561202 8B4D08          mov ecx,[ebp+0x08]
32561205 8901            mov [ecx],eax
32561207 895104          mov [ecx+0x04],edx
3256120A 8BC1            mov eax,ecx
3256120C 5D              pop ebp
3256120D C3              ret
3256120E 90              nop
3256120F 90              nop
cc3250.__ecvt:
32561210 55              push ebp
```



**Zde je návrat z fce. Jak je vidět, fce nic ze zásobníku nemaže.**

# 4) Různé typy volání funkce:

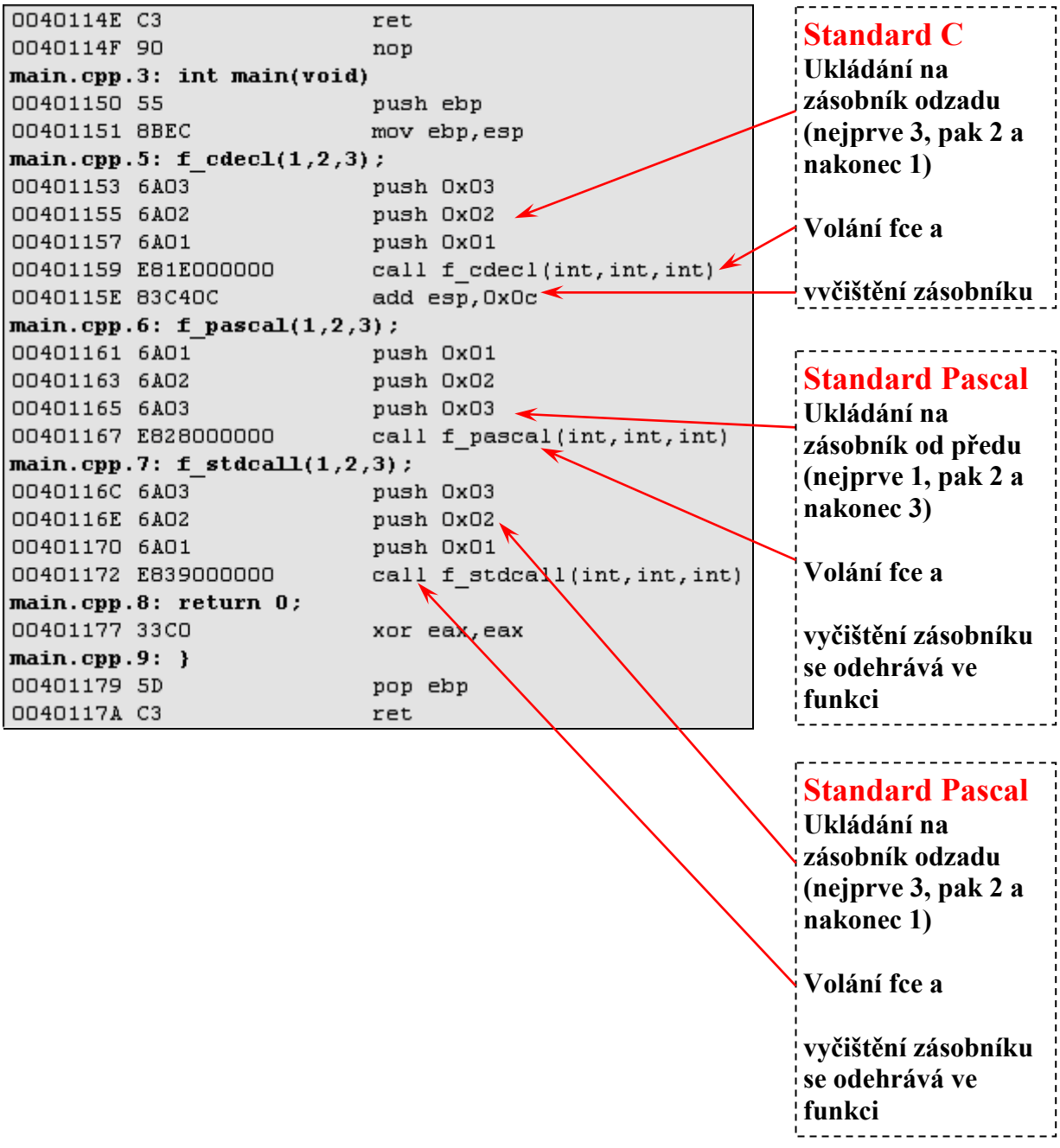
V souboru ***fce.cpp*** jsou definovány tři fce tak, že každá má definovánu jinou konvenci volání.

```
int __cdecl f_cdecl(int a,int b,int c);
int __pascal f_pascal(int a,int b,int c);
int __stdcall f_stdcall(int a,int b,int c);
```

První má standardní Céčkovou konvenci, tzn. parametry se na zásobník ukládají odzadu a volající fce vyčistí zásobník.

Druhá je volaná podle standardu v Pascalu, tzn. parametry se na zásobník ukládají od prvního a volaná fce vyjme parametry ze zásobníku.

Třetí fce je mix obou předchozích, tzn. parametry jsou ukládány odzadu, ale volaná fce je odstraní ze zásobníku.



**Volané funkce:**

```
fce.cpp.3: int __cdecl f_cdecl(int a,int b,int c)
0040117C 55                 push ebp
0040117D 8BEC               mov ebp,esp
fce.cpp.5: return ((a+b+c)/16);
0040117F 8B4508             mov eax,[ebp+0x08]
00401182 8B550C             mov edx,[ebp+0x0c]
00401185 03C2               add eax,edx
00401187 8B4D10             mov ecx,[ebp+0x10]
0040118A 03C1               add eax,ecx
0040118C 85C0               test eax,eax
0040118E 7903               jns +0x03
00401190 83C00F             add eax,0x0f
00401193 C1F804             sar eax,0x04
fce.cpp.6: }
00401196 5D                 pop ebp
00401197 C3                 ret
fce.cpp.8: int __pascal f_pascal(int a,int b,int c)
00401198 55                 push ebp
00401199 8BEC               mov ebp,esp
fce.cpp.10: return ((a+b+c)/16);
0040119B 8B4510             mov eax,[ebp+0x10]
0040119E 8B550C             mov edx,[ebp+0x0c]
004011A1 03C2               add eax,edx
004011A3 8B4D08             mov ecx,[ebp+0x08]
004011A6 03C1               add eax,ecx
004011A8 85C0               test eax,eax
004011AA 7903               jns +0x03
004011AC 83C00F             add eax,0x0f
004011AF C1F804             sar eax,0x04
fce.cpp.11: }
004011B2 5D                 pop ebp
004011B3 C20C00             ret 0x000c
004011B6 90                 nop
004011B7 90                 nop
fce.cpp.13: int __stdcall f_stdcall(int a,int b,int c)
004011B8 55                 push ebp
004011B9 8BEC               mov ebp,esp
fce.cpp.15: return ((a+b+c)/16);
004011BB 8B4508             mov eax,[ebp+0x08]
004011BE 8B550C             mov edx,[ebp+0x0c]
004011C1 03C2               add eax,edx
004011C3 8B4D10             mov ecx,[ebp+0x10]
004011C6 03C1               add eax,ecx
004011C8 85C0               test eax,eax
004011CA 7903               jns +0x03
004011CC 83C00F             add eax,0x0f
004011CF C1F804             sar eax,0x04
fce.cpp.16: }
004011D2 5D                 pop ebp
004011D3 C20C00             ret 0x000c
004011D6 90                 nop
004011D7 90                 nop
___CRTL_VCL_Init:
004011D8 C3                 ret
004011D9 90                 nop
```

**Standard C (cdecl)**
**Fce zásobník nevyprázdní. Vše je na funkce volající.**

**Standard Pascal (pascal)**
**Fce zásobník vyprázdní. Zde je vidět, že ze zásobníku vyjme 12 bytů.**

**MIX (stdcall)**
**Fce opět zásobník vyprázdní. Zde je vidět, že ze zásobníku vyjme 12 bytů.**

# 5) Dumpbin na *crtdll.dll*

```
dumpbin /EXPORTS crtdll.dll > crtdll.txt
```