

# Міністерство освіти та науки України Національний технічний університет України "Київський політехнічний інститут" Факультет інформатики та обчислювальної техніки Кафедра обчислювальної техніки

Лабораторна робота №10 з дисципліни «Системне програмування – 1» на тему: «Використання у проекті С++ модулів на асемблері»

> Виконав: студент 2-го курсу ФІОТ групи ІВ-71 Мазан Я. В. Перевірив: Старший викладач Порєв В. М.

#### Мета:

Навчитися створювати програми на С++ з використанням модулів на асемблері.

## Завдання:

Варіант	Що потрібно обчислити	Розрядність (біт)
9	(1 - A)(B + C)	768

## Код програми:

### main.cpp:

```
#include <iostream>
#include "longop.h"
using namespace std;
int main() {
              unsigned int a[24];
              unsigned int b[24];
              unsigned int c[24];
              unsigned int oneMinusA[25];
              unsigned int bPlusC[25];
              unsigned int result[50];
              /*unsigned int var1[1] = {3};
              unsigned int var2[1] = \{2\};
              unsigned int result[2];
              char text_buffer[5];*/
              fill_n(oneMinusA, 25, 0);
              fill_n(bPlusC, 25, 0);
              fill_n(result, 50, 0);
              unsigned int one[24];
              fill_n(one, 23, 0);
              one[23] = 1;
              /*fill_n(a, 23, 0);
              fill_n(b, 23, 0);
              fill_n(c, 23, 0);
              a[23] = 0;
              b[23] = 0xf;
              c[23] = 1;*/
              unsigned int var = 0x80000000;
              for (int i = 0; i < 24; i++) {
                            a[i] = var;
                            b[i] = 0x80000000;
                            c[i] = 0xffffffff;
                             var += 0x00001000;
              subLongop(oneMinusA, a, one, 24);
                                                        //oneMinusA = 1 - a
              addLongop(bPlusC, c, b, 24);
              mulN_x_N(result, bPlusC, oneMinusA, 25);
              for (int i = 0; i < 50; i++)
                           cout << hex << result[i] << " ";
                                                      result = var1 - var2
  //subLongop(result, var2, var1, 1);
```

//strDec(result, text\_buffer, 2);

```
return 0;
```

# longop.asm:

```
i : resd 1
  j : resd 1
section .text
global mulN_x_N
global addLongop
global subLongop
addLongop:
  push ebp
  mov ebp, esp
  ;push esi
  ;push edx
  push ebx
  ;push ecx
  mov esi, dword [ebp+20]
                               ; length
  mov edx, dword [ebp+16]
                                ; A
                                ; B
  mov ebx, dword [ebp+12]
                               ; C
  mov ecx, dword [ebp+8]
  clc
  pushf
  dec esi
  @addCycle:
    mov eax, dword [edx + esi*4]
    adc eax, dword [ebx + esi*4]
    pushf
    mov dword [ecx+esi*4+4], eax
    dec esi
    cmp\;esi,\,0
    jge @addCycle
    xor eax, eax
    popf
                        ; забираємо наш прапор переносу зі стеку для того, щоби при наступному додаванні його використати
    adc eax, 0
                         ; запис прапору переносу в еах
    mov dword [ecx], eax
  ;pop ecx
  pop ebx
  ;pop edx
  ;pop esi
  leave
                                                                 ; add esp, 16
subLongop:
  push ebp
  mov ebp, esp
  ;push esi
  ;push edx
  push ebx
  ;push ecx
  ;push eax
  mov esi, dword [ebp+20]
                               ; length
                                ; A
  mov edx, dword [ebp+16]
  mov ebx, dword [ebp+12]
                                ; B
  mov ecx, dword [ebp+8]
                               ; C
```

```
pushf
  dec esi
  @subCycle:
    popf
    mov eax, dword [edx + esi*4]
    sbb eax, dword [ebx + esi*4]
    pushf
    mov dword [ecx+esi*4], eax
    dec esi
    cmp\;esi,\,0
    jge @subCycle
    popf
  ;pop eax
  ;pop ecx
  pop ebx
  ;pop edx
  ;pop esi
  leave
                                                                    ; add esp, 16
  ret
mulN_x_N:
                               ; size, A, B, res
  push ebp
  mov ebp, esp
  ;push esi
  push ebx
  ;push edi
  ;push ecx
  ;push eax
  mov esi, dword [ebp + 20]
                                  ; size
  dec esi
                          ; Making pointer to point on start of
  push esi
                           ; save pointer on the last element
  mov dword [i], esi
                              ; i - iterates over A's dwords
  mov dword [j], esi
                                ; j - iterates over B's dwords
                         ; base pointer: edi = i + j
  mov esi, dword [ebp + 16]
                                  ; A
  mov ebx, dword [ebp + 12]
                                   ; B
  mov edi, dword [ebp + 8]
                                  ; result
  @ outer Cycle:
    pop dword [j]
    push dword [j]
    @innerCycle:
      mov ecx, dword [i]
      mov eax, dword [esi + 4*ecx] ; eax = dword [A + i]
      mov ecx, dword [j]
      mul dword [ebx + 4*ecx] ; eax *= dword [B + j]
      add dword [edi + 4*ecx + 4], eax; res = dword [edi + i + j + 4], eax
      adc\ dword\ [edi+4*ecx],\ edx\quad \  ;\ res=dword\ [edi+i+j],\ edx
      push ecx
      pushf
      @addcarryflag:
                               ; spread carry flag over all other digit numbers
         popf
         adc dword [edi + 4*ecx - 4], 0
         pushf
         dec ecx
         cmp\;ecx,\,0
         jge @addcarryflag
      popf
      pop ecx
      mov eax, dword [j]
      dec eax
      mov dword [j], eax
      cmp eax, 0
      jge @innerCycle
    mov eax, dword [i]
```

```
dec eax
mov dword [i], eax
cmp eax, 0
jge @outerCycle

pop esi
;pop eax
;pop ecx
;pop edi
pop ebx
;pop esi

leave
ret ; add esp, 16
```

## Асембльований main.cpp (main.s):

```
.file
                                                                                                      leave
             .intel_syntax noprefix
                                                                                                      .cfi_restore 5
                                                                                                      .cfi_def_cfa 4, 4
             .text
             .section
             .type
                          _ZStL19piecewise_construct, @object
                                                                                                      .cfi_endproc
                         ZStL19piecewise construct, 1
                                                                                         .LFE1730:
            .size
_ZStL19piecewise_construct:
            .zero
                                                                                                      \_ZSt6fill\_nIPjiiET\_S1\_T0\_RKT1\_, .\_ZSt6fill\_nIPjiiET\_S1\_T0\_RKT1\_
                          ZStL8 ioinit
                                                                                                                   . text.\_ZSt12\_niter\_baseIPjET\_S1\_, "axG", @progbits,\_ZSt12\_nite
             .local
                                                                                                      section
                          _ZStL8__ioinit,1,1
                                                                                         r_baseIPjET_S1_,comdat
             .text
                                                                                                      .weak
                                                                                                                   _ZSt12__niter_baseIPjET_S1_
                                                                                                                  _ZSt12__niter_baseIPjET_S1_, @function
             .globl
                         main
                                                                                                      .type
                         main, @function
                                                                                         _ZSt12__niter_baseIPjET_S1_:
             .type
main:
                                                                                         .LFB1841:
.LFB1490:
                                                                                                      .cfi_startproc
             .cfi_startproc
                                                                                                      push
             push
                                                                                                      .cfi_def_cfa_offset 8
            .cfi_def_cfa_offset 8
                                                                                                      .cfi_offset 5, -8
             .cfi_offset 5, -8
                                                                                                      mov
                                                                                                      .cfi_def_cfa_register 5
             .cfi_def_cfa_register 5
                                                                                                      call
                                                                                                                   __x86.get_pc_thunk.ax
                                                                                                      add
                                                                                                                   eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_
            push
                         ebx
             and
                         esp, -16
                                                                                                                   eax, DWORD PTR 8[ebp]
                                                                                                      pop
            sub
                         esp, 352
             .cfi offset 3, -12
                                                                                                      .cfi restore 5
            call
                          __x86.get_pc_thunk.bx
                                                                                                      .cfi_def_cfa 4, 4
             add
                          ebx, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_
                                                                                                      ret
                          eax. DWORD PTR gs:20
                                                                                                      .cfi_endproc
             mov
                                                                                         .LFE1841:
                          DWORD PTR 348[esp], eax
             mov
                                                                                                                   _ZSt12__niter_baseIPjET_S1_, .-_ZSt12__niter_baseIPjET_S1_
                         DWORD PTR 40[esp], 3
                                                                                                                   .text._ZSt10__fill_n_aIPjiiEN9__gnu_cxx11__enable_ifIXsrSt11_
            mov
                                                                                         is\_scalarIT1\_E7\_valueET\_E6\_typeES6\_T0\_RKS4\_,"axG",@progbits,\_ZSt10\_fill\_n\_aIPjiiE
                          DWORD PTR 44[esp], 2
             mov
                          DWORD PTR 20[esp], -2147483648
                                                                                         N9\_gnu\_cxx11\_enable\_ifIXsrSt11\_is\_scalarIT1\_E7\_valueET\_E6\_typeES6\_T0\_RKS4\_, co
                          DWORD PTR 24[esp], 0
            mov
.L3:
                                                                                                      .weak
                          DWORD PTR 24[esp], 23
             cmp
                                                                                         _ZSt10__fill_n_aIPjiiEN9__gnu_cxx11__enable_ifIXsrSt11__is_scalarIT1_E7__valueET_E6__t
                          eax, DWORD PTR 24[esp]
                                                                                         ypeES6_T0_RKS4_
             mov
                          edx, DWORD PTR 20[esp]
             mov
                                                                                                      .type
             mov
                          DWORD PTR 56[esp+eax*4], edx
                          eax, DWORD PTR 24[esp]
                                                                                         _ZSt10__fill_n_aIPjiiEN9__gnu_cxx11__enable_ifIXsrSt11__is_scalarIT1_E7__valueET_E6__t
             mov
                          DWORD PTR 152[esp+eax*4], -2147483648
                                                                                         ypeES6_T0_RKS4_, @function
             add
                          DWORD PTR 20[esp], 4096
                                                                                         _ZSt10__fill_n_aIPjiiEN9__gnu_cxx11__enable_ifIXsrSt11__is_scalarIT1_E7__valueET_E6__t
                         DWORD PTR 24[esp]
                                                                                         ypeES6_T0_RKS4_:
            inc
            jmp
.L2:
                                                                                                      .cfi_startproc
                          DWORD PTR 12[espl, 24
             mov
                                                                                                      push
                          eax, 56[esp]
                                                                                                      .cfi_def_cfa_offset 8
             lea
                          DWORD PTR 8[esp], eax
             mov
             lea
                          eax, 152[esp]
                                                                                                      mov
                                                                                                                   ebp, esp
                         DWORD PTR 4[esp], eax
                                                                                                      .cfi_def_cfa_register 5
             mov
                          eax, 248[esp]
             mov
                          DWORD PTR [esp], eax
                                                                                                      call
                                                                                                                   __x86.get_pc_thunk.ax
```

	call	addLongop@PLT		add	eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_
	mov	DWORD PTR 28[esp], 0		mov	eax, DWORD PTR 16[ebp]
.L5:				mov	eax, DWORD PTR [eax]
	cmp	DWORD PTR 28[esp], 24		mov	DWORD PTR -4[ebp], eax
	jg	.L4		mov	eax, DWORD PTR 12[ebp]
	mov	eax, DWORD PTR 28[esp]	7.40	mov	DWORD PTR -8[ebp], eax
	mov	eax, DWORD PTR 248[esp+eax*4]	.L18:		
	mov	DWORD PTR 4[esp], eax		cmp	DWORD PTR -8[ebp], 0
	mov	eax, DWORD PTR _ZSt4cout@GOT[ebx]		jle	.L17
	mov	DWORD PTR [esp], eax		mov	edx, DWORD PTR -4[ebp]
	call	_ZNSolsEj@PLT		mov	eax, DWORD PTR 8[ebp]
	inc	DWORD PTR 28[esp]		mov	DWORD PTR [eax], edx
	jmp	.L5		dec	DWORD PTR -8[ebp]
.L4:	JP	<del></del>		add	DWORD PTR 8[ebp], 4
.L4.	*****	any DWODD DTD			
	mov	eax, DWORD PTR		jmp	.L18
_ZSt4endilo		nitsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx]	.L17:		
	mov	DWORD PTR 4[esp], eax		mov	eax, DWORD PTR 8[ebp]
	mov	eax, DWORD PTR _ZSt4cout@GOT[ebx]		leave	
	mov	DWORD PTR [esp], eax		.cfi_restore 5	
	call	_ZNSolsEPFRSoS_E@PLT		.cfi_def_cfa 4	4, 4
	mov	DWORD PTR 16[esp], 0		ret	
	lea	eax, 16[esp]		.cfi_endproc	
		-	I EE1042:	.c.i_enuproc	
	mov	DWORD PTR 8[esp], eax	.LFE1842:		
	mov	DWORD PTR 4[esp], 25		.size	
	lea	eax, 248[esp]			
	mov	DWORD PTR [esp], eax	_ZSt10fil	_n_aIPjiiEN9_	_gnu_cxx11enable_ifIXsrSt11is_scalarIT1_E7valueET_E6t
	call	_ZSt6fill_nIPjiiET_S1_T0_RKT1_	ypeES6_T0_	_RKS4_,	
	mov	DWORD PTR 12[esp], 24	_ZSt10_ fil	_n_aIPjiiEN9	_gnu_cxx11enable_ifIXsrSt11is_scalarIT1_E7valueET_E6t
	lea	eax, 56[esp]	ypeES6_T0_	-	
		DWORD PTR 8[esp], eax	JPC250_10_		
	mov			.text	
	lea	eax, 152[esp]		.type	_Z41static_initialization_and_destruction_0ii, @function
	mov	DWORD PTR 4[esp], eax	_Z41statio	_initialization_	and_destruction_0ii:
	lea	eax, 248[esp]	.LFB1980:		
	mov	DWORD PTR [esp], eax		.cfi_startproc	
	call	subLongop@PLT		push	ebp
	mov	DWORD PTR 32[esp], 0		.cfi_def_cfa_	offset 8
.L7:		L-TD		.cfi_offset 5,	
	cmp	DWORD PTR 32[esp], 24		mov	ebp, esp
	-			.cfi_def_cfa_	
	jg	.L6			
	mov	eax, DWORD PTR 32[esp]		push	ebx
	mov	eax, DWORD PTR 248[esp+eax*4]		sub	esp, 20
	mov	DWORD PTR 4[esp], eax		.cfi_offset 3,	-12
	mov	eax, DWORD PTR _ZSt4cout@GOT[ebx]		call	x86.get_pc_thunk.bx
	mov	DWORD PTR [esp], eax		add	ebx, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_
	call	_ZNSolsEj@PLT		cmp	DWORD PTR 8[ebp], 1
	inc	DWORD PTR 32[esp]		jne	L22
		.L7		-	
1.0	jmp	.L/		cmp	DWORD PTR 12[ebp], 65535
.L6:				jne	.L22
	mov	DWORD PTR 16[esp], 0		lea	eax, _ZStL8ioinit@GOTOFF[ebx]
	lea	eax, 16[esp]		mov	DWORD PTR [esp], eax
	mov	DWORD PTR 8[esp], eax		call	_ZNSt8ios_base4InitC1Ev@PLT
	mov	DWORD PTR 4[esp], 5		lea	eax,dso_handle@GOTOFF[ebx]
	lea	eax, 48[esp]		mov	DWORD PTR 8[esp], eax
		DWORD PTR [esp], eax			eax, _ZStL8ioinit@GOTOFF[ebx]
	mov			lea	
	call	_ZSt6fill_nIPjiiET_S1_T0_RKT1_		mov	DWORD PTR 4[esp], eax
	mov	eax, DWORD PTR		mov	eax, DWORD PTR _ZNSt8ios_base4InitD1Ev@GOT[ebx]
_ZSt4endlId	cSt11char_tra	nitsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx]		mov	DWORD PTR [esp], eax
	mov	DWORD PTR 4[esp], eax		call	cxa_atexit@PLT
	1110 4		.L22:		
	mov	eax, DWORD PTR _ZSt4cout@GOT[ebx]		nop	
		eax, DWORD PTR _ZSt4cout@GOT[ebx] DWORD PTR [esp], eax		пор	
	mov mov	DWORD PTR [esp], eax			esp, 20
	mov mov call	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT		add	esp, 20
76-4. "	mov mov call mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR		add pop	ebx
_ZSt4endlIo	mov mov call mov cSt11char_tra	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR itisIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx]		add pop .cfi_restore 3	ebx
_ZSt4endlIo	mov mov call mov cSt11char_tra mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR itisIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx		add pop .cfi_restore 3 pop	ebx ebp
_ZSt4endlId	mov mov call mov cSt11char_tra	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR itisIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx]		add pop .cfi_restore 3	ebx ebp
_ZSt4endlId	mov mov call mov cSt11char_tra mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR itisIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx		add pop .cfi_restore 3 pop	ebx ebp
_ZSt4endHo	mov call mov ccSt11char_tra mov mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR hitsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx DWORD PTR [esp], eax		add pop .cfi_restore 3 pop .cfi_restore 5	ebx ebp
_ZSt4endlId	mov call mov ccst11char_tra mov mov call mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR ditsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT DWORD PTR 12[esp], 1		add pop .cfi_restore 3 pop .cfi_restore 5 .cfi_def_cfa 4 ret	ebx ebp
_ZSt4endHo	mov call mov cSt11char_tra mov mov call mov lea	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR ditsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT DWORD PTR 12[esp], 1 eax, 40[esp]	FE1000-	add pop .cfi_restore 3 pop .cfi_restore 5 .cfi_def_cfa_4	ebx ebp
_ZSt4endlId	mov call mov ccst11char_tra mov mov call mov	DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT edx, DWORD PTR ditsIcEERSt13basic_ostreamIT_T0_ES6_@GOT[ebx] DWORD PTR 4[esp], edx DWORD PTR [esp], eax _ZNSolsEPFRSoS_E@PLT DWORD PTR 12[esp], 1	.LFE1980:	add pop .cfi_restore 3 pop .cfi_restore 5 .cfi_def_cfa 4 ret	ebx ebp

Section			DIJODD DTD 411	7.41	1-141-1141	
				_Z41Static_		
18				GLOBAL		_GLOBALsub_1_main, @function
1908   1908   1908   1918			- *-		suo_i_mani.	
1.00				LI'DIJOI.	cfi startnroc	
Control   Con	1.9.	mov	B WORD I TR Solespi, o		-	
	1201	cmp	DWORD PTR 36[esp], 1			-
Mathematical   Mat			-			
March   Mar						
Second   S			-			
March   Mar		mov				
Mathematical   Math		mov	eax, DWORD PTR _ZSt4cout@GOT[ebx]		call	_x86.get_pc_thunk.ax
MORD PTR (sep)		mov	DWORD PTR [esp], eax		add	
1.50		call	_ZNSolsEj@PLT		mov	DWORD PTR 4[esp], 65535
1.		inc	DWORD PTR 36[esp]		mov	DWORD PTR [esp], 1
mov		jmp	.L9		call	_Z41static_initialization_and_destruction_0ii
	.L8:				leave	
		mov	eax, 0		.cfi_restore 5	
		mov	ecx, DWORD PTR 348[esp]		.cfi_def_cfa 4	1, 4
LiPE 1981  LiPE 1982  LiPE 1983  LiPE 1983		xor	ecx, DWORD PTR gs:20		ret	
		je	.L11		$.cfi\_endproc$	
mov		call	stack_chk_fail_local	.LFE1981:		
	.L11:				.size	_GLOBALsub_I_main,GLOBALsub_I_main
Cl.   Section		mov	ebx, DWORD PTR -4[ebp]			.init_array,"aw"
		leave			.align 4	
					-	
Ret					.section	.textx86.get_pc_thunk.ax,"axG",@progbits,x86.get_pc_thunk.
			1, 4	ax,comdat		
					-	
	I FF4 400	.cti_endproc				
	.LFE1490:		and a section	06		x86.get_pc_thunk.ax, @function
					_tnunk.ax:	
mov eax, DWORD PTR [esp]   mov eax, DWORD PTR [esp]	NIDWET C			.LFD1902;	ofi startproc	
	_IIIFJIIE1_3.				_	
						cus, b words i in [csp]
LFB1730:	ZSt6fill nII					
.cf;_startproc		-,		.LFE1982:	p	
push   ebp   c.fi_def_cfa_offset 8   .globl  x86.get_pc_thunk.bx   .hidden  x86.get_pc_thunk.bx   .h		.cfi startproc			.section	.text. x86.get pc thunk.bx,"axG".@progbits. x86.get pc thunk.
.cfi_def_cfa_offset 8 .cfi_offset 5, -8 mov ebp, esp .cfi_def_cfa_register 5 sub esp, 24 callx86.get_pc_thunk.bx add eax, OFFSET FLAT: GLOBAL_OFFSET_TABLE_ mov eax, DWORD PTR [esp], eax callZSt12_niter_baseIPjET_S1_ mov edx, DWORD PTR [6bp] mov edx, DWORD PTR [2[ebp] mov edx, DWORD PTR [2[ebp]] mov edx, DWORD PTR [4esp], edx mov DWORD PTR [4esp], edx		-		bx,comdat		
.cfi_offset 5, -8 mov ebp, esp .cfi_def_cfa_register 5 sub esp, 24 callx86.get_pc_thunk.ax add eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_ mov eax, DWORD PTR 8[ebp] mov edx, DWORD PTR 8[esp], eax callZS112_niter_baselPjET_S1_ mov edx, DWORD PTR 8[ebp] mov edx, DWORD PTR 8[esp], edx mov edx, DWORD PTR 8[esp], edx indidenstack_chk_fail_local identStack_chk_fail_local identGCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0"  .section .note.GNU-stack,"",@progbits ZSt10_fill_n_alPjiiEN9_gnu_cxx11_enable_iffXsrSt11_is_scalarTT1_E7_valueET_E6_t		-			.globl	_x86.get_pc_thunk.bx
mov ebp, esp .cfi_def_cfa_register 5 sub esp, 24 .callx86.get_pc_thunk.ax add eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_ mov eax, DWORD PTR {ebp} mov DWORD PTR {esp}, eax callZSt12niter_baseIPjET_S1_					-	
x86.get_pc_thunk.bx:  x86.get_pc_thunk.ax  x86.get_pc_thunk.bx:  x86.get_pc_thunk.pc.  x86.						
sub esp, 24 call _x86.get_pc_thunk.ax add eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_ mov eax, DWORD PTR [ebp] mov DWORD PTR [ebp], eax call _ZSt12_niter_baselPjET_S1_ LFE1983: mov edx, DWORD PTR 16[ebp] mov DWORD PTR 16[ebp] mov DWORD PTR 16[ebp] mov dw, DWORD PTR 18[esp], edx mov edx, DWORD PTR 12[ebp] mov DWORD PTR 12[ebp] add eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_ mov ebx, DWORD PTR 16[ebp] hidden _dso_handle hidden _stack_chk_fail_local ident "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0" section .note.GNU-stack,"",@progbits  _ZSt10_fill_n_alPjiiEN9_gnu_cxx11_enable_ifIXsrSt11_is_scalarIT1_E7_valueET_E6_t		.cfi_def_cfa_		x86.get_pc		
add eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_ mov eax, DWORD PTR 8[ebp] mov DWORD PTR [esp], eax call _ZSt12niter_baselPjET_S1_ mov edx, DWORD PTR 16[ebp] .hiddendso_handle .hiddenstack_chk_fail_local .hiddenstack_chk_fail_local .ident "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0" .section .note.GNU-stack,"",@progbits  _ZSt10fill_n_alPjiiEN9gnu_cxx11enable_ifIXsrSt11is_scalarIT1_E7valueET_E6t						
mov eax, DWORD PTR 8[ebp] ret .cfi_endproc call _ZSt12_niter_baselPjET_S1_		call	x86.get_pc_thunk.ax		.cfi_startproc	
mov DWORD PTR [esp], eax .cfi_endproc .call _ZSt12_niter_baselPjET_S1_		add	eax, OFFSET FLAT:_GLOBAL_OFFSET_TABLE_		mov	ebx, DWORD PTR [esp]
call _ZSt12_niter_baselPjET_S1_		mov	eax, DWORD PTR 8[ebp]		ret	
mov edx, DWORD PTR 16[ebp] .hiddendso_handle mov DWORD PTR 8[esp], edx .hiddenstack_chk_fail_local .hiddenstack_chk_fail_local .ident "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0" .section .note.GNU-stack,"",@progbits  _ZSt10_fill_n_aIPjiiEN9_gnu_cxx11_enable_ifIXsrSt11_is_scalarIT1_E7_valueET_E6_t		mov	DWORD PTR [esp], eax		.cfi_endproc	
mov DWORD PTR 8[esp], edx .hiddenstack_chk_fail_local .ident "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0" mov DWORD PTR 4[esp], edx .section .note.GNU-stack,"",@progbits .section .note.GNU-stack,"",@progbits		call	_ZSt12niter_baseIPjET_S1_	.LFE1983:		
mov edx, DWORD PTR 12[ebp] .ident "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0" mov DWORD PTR 4[esp], edx .section .note.GNU-stack,"",@progbits  DWORD PTR [esp], eax call  _ZSt10_fill_n_aIPjiiEN9_gnu_cxx11_enable_ifIXsrSt11_is_scalarIT1_E7_valueET_E6_t		mov	edx, DWORD PTR 16[ebp]		.hidden	dso_handle
mov DWORD PTR 4[esp], edx .section .note.GNU-stack,"",@progbits mov DWORD PTR [esp], eax call  _ZSt10_fill_n_aIPjiiEN9_gnu_cxx11_enable_ifIXsrSt11_is_scalarIT1_E7_valueET_E6_t		mov	DWORD PTR 8[esp], edx		.hidden	stack_chk_fail_local
mov DWORD PTR [esp], eax call  _ZSt10fill_n_aIPjiiEN9gnu_cxx11enable_ifIXsrSt11is_scalarIT1_E7valueET_E6t		mov			.ident	"GCC: (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0"
call  _ZSt10fill_n_aIPjiiEN9gnu_cxx11enable_iffXsrSt11is_scalarIT1_E7valueET_E6t		mov	DWORD PTR 4[esp], edx		.section	.note.GNU-stack,"",@progbits
_ZSt10fill_n_aIPjiiEN9gnu_cxx11enable_iffXsrSt11is_scalarIT1_E7valueET_E6t		mov	DWORD PTR [esp], eax			
		call				
ypeES6_T0_RKS4_		-	_gnu_cxx11enable_ifIXsrSt11is_scalarIT1_E7valueET_E6t			
	ypeES6_T0_	RKS4_				

# Makefile:

CC=gcc CPLUS=g++ ASM=nasm LD=g++

#File types to compile

## Результати виконання програми:

```
yan@yan-Latttude-5580:~/Desktop/4 cemecrp/Assembly/lab 10/c-asm$ make
g++ -march=1686 -m32 -o prog main.o longop.o
yan@yan-Latttude-5580:~/Desktop/4 cemecrp/Assembly/lab 10/c-asm$ ./prog
obfffffff 7fffe7ff 3fffc7fe ffff9ffe bfff6ffe 7fff37fe 3ffef7fd fffeaffd bffesffd 7ffe07fd 3ffda7fc fffd3ffc bffccffc 7ffc57fc 3ffbd7fb fffb4ffb bffabffb 7
ffa27fb 3ff987fa fff887fa 7ff877fa 3ff6b7f9 fff5effb 3ff75fb 7ff777fb bff797fb ff79ffc 3ff2effc 7ff827fc bff867fc fff8affd 3ff8fffd 7ff957fd bff9
b7fd fffa1ffe 3ffa8ffe 7ffb07fe bffb87fe fffc0fff 3ffc9fff 7ffd37ff bffdd7ff fffe8000 3fff3000 7fffe800 c000a801 16fff 0 yan@yan-Latttude-5580:~/Desktop/4 c
emecrp/Assembly/lab 10/c-asm$
```

#### Висновок:

Під час виконання даної лабораторної роботи були закріплені навички використання процедур написаних на асемблері в програмах С++. Під час реалізації розрахунку виразів були виявлені фатальні помилки в процедурах, але всі вони були вдало усунені.