#include <windows.h>

#include <iostream>

#include <eh.h>

using namespace std;

void se\_trans\_func(unsigned code, EXCEPTION\_POINTERS \*)

{

throw code;

}

int main()

{

int a = 10, b = 0;

// устанавливаем функцию-транслятор

\_set\_se\_translator(se\_trans\_func);

// перехватываем структурное исключение средствами C++

try

{

a /= b; // ошибка, деление на ноль

}

catch (unsigned code)

{

cout << "Exception code = " << hex << code << endl;

}

return 0;

}

Приведем в листинге 2.14 программу, которая выбрасывает структуру типа EXCEPTION\_POINTERS как исключение языка программирования C++.

#include <windows.h>

#include <iostream>

#include <eh.h>

using namespace std;

void se\_trans\_func(unsigned code, EXCEPTION\_POINTERS \*info)

{

EXCEPTION\_RECORD er;

CONTEXT c;

EXCEPTION\_POINTERS ep = { &er, &c };

er = \*(info->ExceptionRecord);

c = \*(info->ContextRecord);

throw ep;

}

int main()

{

int a = 10, b = 0;

// устанавливаем функцию-транслятор

\_set\_se\_translator(se\_trans\_func);

// перехватьшаем структурное исключение средствами C++

try

{

a /= b; // ошибка, деление на ноль

}

catch (EXCEPTION\_POINTERS ep)

{

cout << "Exception code = " << hex << ep.ExceptionRecord->ExceptionCode << endl;

}

return 0;

}

**10.**

#include "stdafx.h"

#include <windows.h>

#include <iostream>

#include <eh.h>

using namespace std;

int main()

{

int a = 10;

int b = 0;

\_try

{

a = a / b;

}

\_finally

{

// если блок try закончился нормально

if (!AbnormalTermination())

{

cout << "try was done." << endl;

}

else

// иначе нечего освобождать

cout << "try wasn't done." << endl;

}

return 0;

}

**Вывод**

Существуют небольшие отличия механизма структурной обработки исключений в Windows от механизма обработки исключений, принятого в языке программирования C++.

Это связано с тем, что:

* механизм структурной обработки исключений был разработан раньше, чем принят стандарт языка C++;
* механизм структурной обработки исключений в отличие от языка программирования C++ ориентирован не только на обработку программных исключений, но и на обработку аппаратных исключений.

В SEH исключение рассматривается как ошибка, происшедшая при выполнении программы. В языке программирования C++ используется более абстрактный подход и исключение рассматривается как объект произвольного типа, который может выбросить программа, используя оператор throw.

В свою очередь обработчик исключения catch может рассматриваться как функция с одним параметром, которая выполняется только в том случае, если тип ее параметра соответствует типу выброшенного исключения.

**Список литературы**

1. Документация по Visual C++ [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/cpp/ (Дата обращения: 23.11.17).
2. Программирование [Электронный ресурс]. – Режим доступа: https://prog-cpp.ru/ (Дата обращение: 23.11.17).
3. Сайт о программировании [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/tutorial/2.14.php (Дата обращение: 23.11.17).