**14** Reference Parameters and Variables in C++

Reference Parameters

1 // ex1401.cpp C++ reference parameters

& indicates a is a

reference parameter

2 #include <iostream>

3 using namespace std;

4 int x = 5;

5 void f(int &a)

6 {

7 a = a + 1; // adds 1 to x

8 }

9 //===================

10 int main()

11 {

12 cout << x << endl;

13 f(x);

14 cout << x << endl;

15 return 0;

16 }

Pass by Address

1 // ex1402.cpp Explicitly passing and dereferencing addresses

2 #include <iostream>

3 using namespace std;

4 int x = 5;

5 void f(int \*a)

6 {

7 \*a = \*a + 1; // adds 1 to x

8 }

9 //===================

10 int main()

11 {

12 cout << x << endl;

13 f(&x);

14 cout << x << endl;

15 return 0;

16 }

Using Reference Parameter

1 ; ex1401.a C++ reference parameters

2 startup: bl main

3 halt

4 ;==============================================================

5 ; #include <iostream>

6 ; using namespace std;

7 x: .word 5 ; int x = 5;

8

9 @f$ri: push lr ; void f(int &a)

10 push fp ; {

11 mov fp, sp

12

13 ldr r0, fp, 2 ; a = a + 1;

14 ldr r0, r0, 0

Dereferencing a

15 add r0, r0, 1

16 ldr r1, fp, 2

Dereferencing a

17 str r0, r1, 0

18

19 mov sp, fp ; }

20 pop fp

21 pop lr

22 ret

23 ;==============================================================

24 main: push lr ; int main()

25 push fp ; {

26 mov fp, sp

27

28 ld r0, x ; cout << x << endl;

29 dout

Passing address of x

30 nl

31

32 lea r0, x ; f(x);

33 push r0

34 bl @f$ri

35 add sp, sp, 1

36

37 ld r0, x ; cout << x << endl;

38 dout

39 nl

40

41 mov r0, 0 ; return 0;

42 mov sp, fp

43 pop fp

44 pop lr

45 ret

46 ; }

Reference Variables

A *reference variable* is a variable that is an *alias* (i.e., an alternate name) for another variable.

int &xr = x;

xr = 5; // xr alias for x so 5 is assigned to x

A reference variable is just a pointer variable that points to the variable for which it is an alias. Thus, the declaration above creates xr pointing to x:

xr x

A reference variable is not like a regular pointer variable in three respects:

1. Once created, it cannot be modified. In other words, it is a constant pointer.
2. Wherever a reference variable is used, the compiler automatically generates code that dereferences it. In contrast, a regular pointer is dereferenced only if it is preceded by an asterisk.
3. A reference variable can never have the value NULL.

Reference Variable Example

1 ; ex1402.a C++ reference variables

2 startup: bl main

3 halt

4 ;==============================================================

5 ; #include <iostream>

6 ; using namespace std;

7 x: .word 5 ; int x = 5;

8 xr: .word x ; int &xr = x;

9

10 main: push lr ; int main()

11 push fp ; {

12 mov fp, sp

13

14 mov r0, 7 ; int y = 7;

15 push r0

16 add r0, fp, -1 ; int &yr = y;

17 push r0

18

19 ld r0, x ; cout << x << endl;

20 dout

21 nl

22

23 ld r0, xr ; cout << xr << endl;

24 ldr r0, r0, 0

Dereferencing xr

25 dout

26 nl

27

28 ldr r0, fp, -1 ; cout << y << endl;

29 dout

30 nl

31

32 ldr r0, fp, -2 ; cout << yr << endl;

33 ldr r0, r0, 0

Dereferencing yr

34 dout

35 nl

36

37 mov r0, 0 ; return 0;

38 mov sp, fp

39 pop fp

40 pop lr

41 ret

42 ; }