

```
entry:
%retval = alloca i32, align 4
%A = alloca [10 x i32], align 16
%B = alloca [10 x i32], align 16
%i = alloca i32, align 4
%j = alloca i32, align 4
store i32 0, i32* %retval, align 4
%0 = bitcast [10 x i32]* %A to i8*
call void @llvm.memcpy.p0i8.p0i8.i64(i8* align 16 %0, i8* align 16 bitcast
... ([10 x i32]* @_const.main.A to i8*), i64 40, i1 false)
%1 = bitcast [10 x i32]* %B to i8*
call void @llvm.memset.p0i8.i64(i8* align 16 %1, i8 0, i64 40, i1 false)
store i32 0, i32* %j, align 4
store i32 0, i32* %i, align 4
%2 = load i32, i32* %j, align 4
%idxprom = sext i32 %2 to i64
%arrayidx = getelementptr inbounds [10 x i32], [10 x i32]* %A, i64 0, i64
... %idxprom
%3 = load i32, i32* %arrayidx, align 4
%mul = mul nsw i32 %3, 23
%var = alloca i32, align 16
store i32 %mul, i32* %var, align 4
%4 = load i32, i32* %j, align 4
%idxprom5 = sext i32 %4 to i64
%arrayidx6 = getelementptr inbounds [10 x i32], [10 x i32]* %A, i64 0, i64
... %idxprom5
%5 = load i32, i32* %arrayidx6, align 4
%var1 = alloca i32, align 16
store i32 %5, i32* %var1, align 4
br label %for.cond
```

```
for.cond:
%6 = load i32, i32* %i, align 4
%cmp = icmp slt i32 %6, 10
br i1 %cmp, label %for.body, label %for.end, !prof !34
```

T

F

```
for.body:
%7 = load i32, i32* %i, align 4
%fix = load i32, i32* %var, align 4
%add = add nsw i32 %fix, %7
%8 = load i32, i32* %i, align 4
%idxprom1 = sext i32 %8 to i64
%arrayidx2 = getelementptr inbounds [10 x i32], [10 x i32]* %B, i64 0, i64
... %idxprom1
store i32 %add, i32* %arrayidx2, align 4
%9 = load i32, i32* %i, align 4
%idxprom3 = sext i32 %9 to i64
%arrayidx4 = getelementptr inbounds [10 x i32], [10 x i32]* %B, i64 0, i64
... %idxprom3
%10 = load i32, i32* %arrayidx4, align 4
%fix2 = load i32, i32* %var1, align 4
%add7 = add nsw i32 %10, %fix2
%add8 = add nsw i32 %add7, 7
%11 = load i32, i32* %i, align 4
%idxprom9 = sext i32 %11 to i64
%arrayidx10 = getelementptr inbounds [10 x i32], [10 x i32]* %B, i64 0, i64
... %idxprom9
store i32 %add8, i32* %arrayidx10, align 4
%12 = load i32, i32* %i, align 4
%rem = srem i32 %12, 8
%cmp11 = icmp eq i32 %rem, 0
br i1 %cmp11, label %if.then, label %if.end, !prof !35
```

T

F

```
if.then:
%13 = load i32, i32* %i, align 4
%14 = sext i32 %13 to i64
%15 = getelementptr inbounds [10 x i32], [10 x i32]* %A, i64 0, i64 %14
%16 = load i32, i32* %15, align 4
%17 = mul nsw i32 %16, 23
store i32 %17, i32* %var, align 4
%18 = sext i32 %13 to i64
%19 = getelementptr inbounds [10 x i32], [10 x i32]* %A, i64 0, i64 %18
%20 = load i32, i32* %19, align 4
store i32 %20, i32* %var1, align 4
br label %if.end
```

```
if.end:
%21 = load i32, i32* %i, align 4
%idxprom12 = sext i32 %21 to i64
%arrayidx13 = getelementptr inbounds [10 x i32], [10 x i32]* %B, i64 0, i64
... %idxprom12
%22 = load i32, i32* %arrayidx13, align 4
%call = call i32 (i8*, ...) @printf(i8* noundef getelementptr inbounds ([4 x
... i8], [4 x i8]* @.str, i64 0, i64 0), i32 noundef %22)
br label %for.inc
```

```
for.inc:
%23 = load i32, i32* %i, align 4
%inc = add nsw i32 %23, 1
store i32 %inc, i32* %i, align 4
br label %for.cond, !llvm.loop !36
```

```
for.end:
ret i32 0
```

CFG for 'main' function