

```
entry:
%0 = alloca i64, align 8
%retval = alloca i32, align 4
%a = alloca i64, align 8
%sum = alloca i64, align 8
%i = alloca i32, align 4
store i32 0, i32* %retval, align 4
store i64 0, i64* %a, align 8
store i64 0, i64* %sum, align 8
store i32 0, i32* %i, align 4
%1 = load i64, i64* %a, align 8
%2 = load i64, i64* %a, align 8
%3 = mul nsw i64 %1, %2
store i64 %3, i64* %0, align 8
br label %for.cond
```

```
for.cond:
%4 = load i32, i32* %i, align 4
%cmp = icmp slt i32 %4, 10000000
br i1 %cmp, label %for.body, label %for.end, !prof !34
```

T	F
---	---

```
for.body:
%5 = load i32, i32* %i, align 4
%rem = srem i32 %5, 20000
%cmp1 = icmp eq i32 %rem, 0
br i1 %cmp1, label %if.then, label %if.else, !prof !35
```

T	F
---	---

```
if.then:
%6 = load i32, i32* %i, align 4
%conv = sext i32 %6 to i64
store i64 %conv, i64* %a, align 8
%7 = load i64, i64* %a, align 8
%8 = load i64, i64* %a, align 8
%9 = mul nsw i64 %7, %8
store i64 %9, i64* %0, align 8
br label %if.end
```

```
if.else:
%10 = load i64, i64* %0, align 8
%11 = load i64, i64* %sum, align 8
%add = add nsw i64 %11, %10
store i64 %add, i64* %sum, align 8
br label %if.end
```

```
if.end:
br label %for.inc
```

```
for.inc:
%12 = load i32, i32* %i, align 4
%inc = add nsw i32 %12, 1
store i32 %inc, i32* %i, align 4
br label %for.cond, !llvm.loop !36
```

```
for.end:
%13 = load i64, i64* %sum, align 8
%call = call i32 @printf(i8* noundef getelementptr inbounds ([14
... x i8], [14 x i8]* @.str, i64 0, i64 0), i64 noundef %13)
ret i32 0
```

CFG for 'main' function