Fixing ReadParquet.cpp Compilation Errors

Problem Summary
Your code defines explicit specializations of the template member function `ColReadOp::read()` **inside the class body**. C++ requires explicit specializations to be at **namespace scope**, not inside the class. Because of this, the compiler falls back to instantiating the generic template, leading to type mismatch errors such as converting `double*` to `float*`, `long int*` to `int*`, and `double*` to `FixedLenByteArray*`.

Root Causes
1. Illegal in-class explicit specializations for:
- `arrow::Int32Type`
- `arrow::UInt32Type`
- `arrow::FloatType`
- `arrow::DoubleType`
- `arrow::Decimal128Type`

2. Type mismatches when calling `ReadBatch(...)`:
- `arrow::FloatType` expects `float*`, but Chapel buffer is `double*`.
- `arrow::Int32Type` expects `int*`, but Chapel buffer is `int64_t*`.
- `arrow::UInt32Type` expects `unsigned int*`, but Chapel buffer is `uint64_t*`.
- `arrow::Decimal128Type` expects `FixedLenByteArray*`, but Chapel buffer is `double*`.

3. Minor bug: `startIdx -= reader->Skip(*startIdx);` (missing `*` on LHS).

Solutions
---
1. **Remove in-class specializations** from `struct ColReadOp`.
2. **Re-add them at namespace scope** (after the class definition), e.g.:

```
template<> inline int64_t ColReadOp::read() {
return _readShortIntegral>();
}
```

Do the same for `UInt32Type`, `FloatType`, `DoubleType`, and `Decimal128Type`.

3. **FloatType specialization**: Read into a temporary `float[]`, then cast each to `double` before storing in the Chapel buffer.

4. **Int32/UInt32 specializations**: Use `_readShortIntegral>()` helper to safely widen to 64-bit Chapel types.

5. **Decimal128 specialization**: Read into `FixedLenByteArray`, then convert via `arrow::Decimal128::FromBigEndian(value.ptr, numbytes)` and store as `double`.

6. **Fix typo**: Use `*startIdx -= reader->Skip(*startIdx);`.

7. **Ensure NAN availability**: Include ``.

Why This Fixes Errors
- Moves specializations to legal scope → resolves “explicit specialization in non-namespace scope” errors.
- Ensures proper type promotion/copy → resolves pointer type mismatch errors.

- Correct Decimal handling avoids treating bytes as doubles directly.

Next Steps

- Apply the diff as described above.
- Verify with test files containing Int32, UInt32, Float, Double, and Decimal128 columns.
- Confirm Chapel buffer receives correctly promoted values.