Floating Point Guidelines

Guidelines

Guideline 1:

> «Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

Guideline 1 – Example

Guideline 1:

«Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

Example:

This is `false`

```
float a = 1.05f;
if (100*a == 105.0f)
    std::cout << "no output\n";
```

Guideline 1 – Example

Guideline 1:

«Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

This is `false`

Example:

```cpp
float a = 1.05f;
if (100*a == 105.0f)
    std::cout << "no output\n";
```

Problem:

`1.05f` not representable

Guideline 1 – Example

Guideline 1:

«Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

This is `false`

Example:

```
float a = 1.05f;
if (100*a == 105.0f)
    std::cout << "no output\n";
```

Problem:

`1.05f` not representable

$$1.05 = 1.00001100110011001100110011001\ldots \cdot 2^0$$
$$\text{(rounding)} \rightarrow 1.049999995231\ldots = 1.00001100110011001100110 \cdot 2^0$$

24bit

Guidelines

Guideline 1:

«Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes**!»

Guideline 2 – Example

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes**!»

Example:

```cpp
float a = 67108864.0f + 1.0f;

if (a > 67108864.0f)
    std::cout << "This is not output ... \n";
```

Guideline 2 – Example

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes**!»

Example:

Problem:

Significand too short

```cpp
float a = 67108864.0f + 1.0f;

if (a > 67108864.0f)
    std::cout << "This is not output ... \n";
```

Guideline 2 – Example

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes**!»

Example:

Problem:

Significand too short

```
float a = 67108864.0f + 1.0f;

if (a > 67108864.0f)
    std::cout << "This is not output ... \n";
```

$$\begin{array}{rcl}
& & \overbrace{}^{24\text{bit}} \\
67108864 & = & 1.00000000000000000000000 \cdot 2^{26} \\
+1 & = & 0.00000000000000000000001 \cdot 2^{26} \\
\hline
67108865 & = & 1.00000000000000000000001 \cdot 2^{26}
\end{array}$$

Guideline 2 – Example

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes**!»

Example:

Problem:

Significand too short

```
float a = 67108864.0f + 1.0f;

if (a > 67108864.0f)
    std::cout << "This is not output ... \n";
```

$$
\begin{array}{rl}
\overset{\text{24bit}}{67108864} = & 1.00000000000000000000000 \cdot 2^{26} \\
+1 = & 0.00000000000000000000001 \cdot 2^{26} \\
\hline
67108865 = & 1.00000000000000000000001 \cdot 2^{26} \\
\text{(rounding)} \rightarrow 67108864 = & 1.00000000000000000000000 \cdot 2^{26}
\end{array}
$$

Guidelines

Guideline 1:

«Do **not** test two floating point numbers for **equality**, if at least one of them was rounded before.»

Guideline 2:

«**Avoid** the **addition** of numbers of extremely **different sizes!**»

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes!**»

Guideline 3 – Example

Guideline 3:

> «**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- Consider sequence $x_{n+1} = 6x_n - 1$

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- Consider sequence $x_{n+1} = 6x_n - 1$
- Computing some sequences for given x_0:

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- Consider sequence $x_{n+1} = 6x_n - 1$
- Computing some sequences for given x_0:
  - e.g. $x_0 = 1$ → $x_1 = 5$, $x_2 = 29$, $x_3 = 173$, …

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- Consider sequence $x_{n+1} = 6x_n - 1$
- Computing some sequences for given x_0:
  - e.g. $x_0 = 1$ $\rightarrow$ $x_1 = 5,\ \ x_2 = 29,\ \ x_3 = 173,\ \ ...$
  - e.g. $x_0 = 0.2$ $\rightarrow$ $x_1 = 0.2,\ \ x_2 = 0.2,\ \ x_3 = 0.2,\ \ ...$

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- Consider sequence $x_{n+1} = 6x_n - 1$

- Computing some sequences for given x_0:
  - e.g. $x_0 = 1$ → $x_1 = 5, \quad x_2 = 29, \quad x_3 = 173, \quad …$
  - e.g. $x_0 = 0.2$ → $x_1 = 0.2, \quad x_2 = 0.2, \quad x_3 = 0.2, \quad …$

C++ claims

$x_{14} \approx 622.982$

Guideline 3 – Example

<div style="border: 1px solid green; background-color: #e8f3d8;">

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

</div>

Example:

- What went wrong?

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- ## What went wrong?
  - `float` represents 0.2 as 0.20000000298...
  - Thus: $6 \cdot x_0 - 1 \neq 1.2 - 1$

Guideline 3 – Example

Example:

- ## What went wrong?
  - `float` represents 0.2 as 0.20000000298...
  - Thus: $6 \cdot x_0 - 1 \neq 1.2 - 1$ but rather:
    $$x_1 = 0.20000004768\ldots$$
    $$x_2 = 0.20000028610\ldots$$
    $$x_3 = 0.20000171661\ldots$$
    $$\vdots$$

Guideline 3 – Example

Guideline 3:

«**Avoid** the **subtraction** of numbers of **similar sizes**!»

Example:

- ## What went wrong?
  - `float` represents 0.2 as 0.20000000298…
  - Thus: $6 \cdot x_0 - 1 \neq 1.2 - 1$ but rather:

$$x_1 = 0.2000000\mathbf{4768}\ldots$$
$$x_2 = 0.200000\mathbf{28610}\ldots$$
$$x_3 = 0.20000\mathbf{171661}\ldots$$
$$\vdots$$

Note how error increases!