小数をどう表すか

- 仮数×$2^{\text{指数}}$ の形式、すなわち仮数部 (mantissa, significand) と指数部 (exponent) に分けて表現する

156

正規化 (normalization)

- 仮数を“1.・・・・”という形式で表現することにする
- 正規化するときに、小数点を動かすので“浮動小数点 (floating point) ”
- いつも“1.”の部分は共通なので、残りの部分を仮数部で表す

157

符号

- 仮数、指数ともに符号が必要だが、1つの数の表現の中に2つも符号があると一目でどんな数なのかわかりにくい
  ⇒ 仮数のみに符号をつける
- 最左ビットを仮数の符号とする
  ⇒ すぐその数の正負が分かる

158

げたばき表示

- 指数部は、00‥0を負の最大値、111‥1を正の最大値とする
  ⇒ げたばき表示 (biased notation) と呼ぶ
  例 0111をゼロにするのは7のげたばき
- 指数部は符号ビットのすぐ右におく
  ⇒ 数の並べ替えをするときに便利
- 結果として
  $(-1)^{s} \times (1 + \text{仮数}) \times 2^{(\text{指数}-\text{げた})}$
  という数を表現している

159

ゼロは例外

- ゼロは例外で符号0、指数部0、仮数部0、すなわち000・・・・0（オール0）とする

160

3.5_{10}の浮動小数点表示

$3.5_{10} = 11.1_2 = 1.11 \times 2^1$

となるので、指数部3ビット（2^2のげたばき）、仮数部4ビットで表現すると

0 1011100

符号　　指数部　　仮数部

161

練習

- 指数部4ビット(7のげたばき)、仮数部11ビットとして、
  ①－0.75を表現しなさい
  ②1100101000000000はどんな数か
- xとsumを float で宣言して，
  for (x=0.1, sum=0.0, i=1; i <= 100; i++)
  sum = sum+x;
  を実行すると，sum が10.0にならないことがある．この理由を考えなさい．

162

現状

- IEEE754では、32ビットを指数部8ビット(127のげたばき)と仮数部23ビットとしている
- 倍精度では、64ビットを指数部11ビット(1023のげたばき)と仮数部52ビットにしている
  ⇒ 絶対値で最小値　2.0×10^{-308}、最大値　2.0×10^{308}が表せる

163

1.0_{10}をdouble型で表すと

$1.0_{10}=1.0_2 \times 2^0$

0 01111111111 000.....0

符号 指数部 仮数部

$03ff0000000000000_{16}$

164

オーバフロー、アンダフロー

- 正規化して、指数部の上限を超えたらオーバフロー、0に近すぎて表現できなければアンダフロー
  ⇒ 演算異常の割込み
  (アンダフローは0にして続行することも多いが、その種の割込みの処理をどうするかをきちんとプログラムで処理しておかないと移植時などに動かなくなる)

165

表現の幅

- 指数部3ビット、仮数部4ビットで表現できる浮動小数点数
  正の最大値　$0.37_8 \times 2^4$
  正の最小値　$0.1_8 \times 2^{-5}$
  負の最大値　$0.74_8 \times 2^3$
  負の最小値　$0.4_8 \times 2^{-4}$

166