【１】数値データの表現形式　　　　　　　　　　　　　　　　　　　　　　　…テキストP.（　　　）

[①　　　　　　　　]

数値データ

（2進数）

：整数表現

[②　　　　　　　　]

：実数表現

※補足

**実数**：プラスマイナス・０・整数・分数・小数、有理数やルートや円周率などの無理数全て含む

**整数**：-2、-1、0、1、2…という数

**自然数**：整数の内、正の数のみ。1、2、3…という数

【２】固定小数点数　　　　　　　　　　　　　　　　　　　　　　　　　　　…テキストP.（　　　）

　固定小数点数の表現形式は1語を決められた大きさ（8ビット、16ビット、32ビットなど）で表現される。このため、一定の大きさの数値を表現しようとすると、[①　　　　　　　　　](桁あふれ)と呼ばれる問題が発生する。

例）8ビット固定小数点数の表現形式

　　（＋１８）１０を固定小数点数で表現すると…

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ０ | ０ | ０ | １ | ０ | ０ | １ | ０ |

　　　　↑　　　　　　　　　　　　　　　　　　　　　　　　　　　▲

　　　符号ビット　　　　　　　　　　　　　　　　　　　　　　小数点の位置

　　　※小数点の位置は、必ず最も右とは決まっていない。

【３】負数の表現方法（2の補数方式）　　　　　　　　　　　　　　　　　　…テキストP.（　　　）

　人間の世界では、負の値を表現するのに“－”（マイナス）記号を用いるが、コンピュータの世界では、マイナスを表現するのに０と１しかないため、これらの組合せ（ビットパターン）をもって負の値を表現しなければならない。この表現方法として、[①　　　　　　　　]というやり方を用いる。

　次の例1、例2でコンピュータ内部での負数の表現、2の補数のやり方について確認しよう！

１．8ビットの2の補数方式で、（－１８）10を表現するには…

（１）絶対値の2進数を求める。 　　　　　　（１８）10＝（０００１ ００１０）2

（２）この結果を反転させて、＋１を加える。　　　０００１ ００１０

　　　　　　　　　　　　　　　　　　　　↓（反転）

　　　　　　　　　　　　　　　　　　　　　　　　１１１０ １１０１

　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　 ＋１

　　　　　　　　　　　　　　　　　　　　　　　　１１１０ １１１０

　∴　解答は、（－１８）10＝（１１１０ １１１０）2 となる。

※負数を2の補数方式で表現した場合、**必ず先頭ビット（符号ビット）は「１」**となる！

２．負の表現（１１１０ １１１０）を、10進数に変換するには…

（１）符号ビットが1なので、負数と判断！　　　　１１１０ １１１０

　　　　　　　　　　　　　　　　　　　　　　　　　　　　↓（反転）

（２）この数を反転させて、＋１を加える。　　　　０００１ ０００１

　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　 ＋１

　　　　　　　　　　　　　　　　　　　　　　　　０００１ ００１０

（３）（２）で求めた値の絶対値を確認する。　　　　　　　　　（１８）10

（４）その値にマイナスをつける。

∴　解答は、（１１１０ １１１０）2＝（－１８）10 となる。

≪例題≫　次の問題について答えよ。なお、負数は２の補数の８ビットで表現している。

①（－10）10＝（　　　　　　　　　　）2

②（－81）10＝（　　　　　　　　　　）2

③（－123）10＝（　　　　　　　　　　）2

④（1111 1011）2＝（　　　　　　）10

⑤（1100 0110）2＝（　　　　　　）10

⑥（1111 1111）2＝（　　　　　　）10

【４】表現可能な範囲（２の補数方式における）　　　　　　　　　　　　　　…テキストP.（　　　）

１．論理データ：自然数の正の数のみの表現

例1　符号ビットを考えない場合（論理データ）

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 最大値 | １ | １ | １ | １ | １ | １ | １ | １ | ：（＋２５５）10 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 最小値 | ０ | ０ | ０ | ０ | ０ | ０ | ０ | ０ | ：（０）10 |

よって、論理データの場合の表現可能な範囲は次の通りとなる。

**０～＋２５５**（[①　　　　　　　　　　　]）

２．算術データ：整数の正と負の数の表現

例2　符号ビットを考えた場合（算術データ）

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 最大値 | ０ | １ | １ | １ | １ | １ | １ | １ | ：（＋１２７）10 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 最小値 | １ | ０ | ０ | ０ | ０ | ０ | ０ | ０ | ：（－１２８）10 |

よって、算術データの場合の表現可能な範囲は次の通りとなる。

**－１２８～＋１２７**（[②　　　　　　　　　　　　　]）

≪範例≫

負数を２の補数で表現する固定小数点表示法において，ｎビットで表現できる整数の範囲はどれか。ここで，小数点の位置は最下位ビットの右とする。

ア　－２ｎ～２ｎ－１ 　　　 イ　－２ｎ－１－1～２n―１

ウ　－２ｎ－１～２ｎ－１－１ エ　－２ｎ－１～２ｎ－１

≪解答≫　ウ

負数を２の補数で表現する固定小数点表示法では、ｎビットで表現できる最大値を２ｎ－１－１で表すことができ、最小値を－２ｎ－１で表すことができます。

具体的に数値を当てはめて考えると、たとえば、８ビットの最大値は、(01111111)２であり、これを10進数に変換すると、127（２８－１－１）となります。８ビットの最小値は、(10000000)２であり、これを10進数に変換すると、－128（－２８－１）となります。

【補足】表現可能な範囲；最小値・最大値のビットパターン

|  |  |  |  |
| --- | --- | --- | --- |
| 10進数 | １語＝４ビット | １語＝６ビット | １語＝８ビット |
| －１２８ |  |  | **１０００００００** |
| －１２７ |  |  | １００００００１ |
| －１２６ |  |  | １０００００１０ |
| ～ |  |  | ～ |
| －３２ |  | **１０００００** | １１１０００００ |
| －３１ |  | １００００１ | １１１００００１ |
| －３０ |  | １０００１０ | １１１０００１０ |
| ～ |  | ～ | ～ |
| －８ | **１０００** | １１１０００ | １１１１１０００ |
| －７ | １００１ | １１１００１ | １１１１１００１ |
| －６ | １０１０ | １１１０１０ | １１１１１０１０ |
| －５ | １０１１ | １１１０１１ | １１１１１０１１ |
| －４ | １１００ | １１１１００ | １１１１１１００ |
| －３ | １１０１ | １１１１０１ | １１１１１１０１ |
| －２ | １１１０ | １１１１１０ | １１１１１１１０ |
| －１ | １１１１ | １１１１１１ | １１１１１１１１ |
| ０ | ００００ | ００００００ | ００００００００ |
| ＋１ | ０００１ | ０００００１ | ０００００００１ |
| ＋２ | ００１０ | ００００１０ | ００００００１０ |
| ＋３ | ００１１ | ００００１１ | ００００００１１ |
| ＋４ | ０１００ | ０００１００ | ０００００１００ |
| ＋５ | ０１０１ | ０００１０１ | ０００００１０１ |
| ＋６ | ０１１０ | ０００１１０ | ０００００１１０ |
| ＋７ | **０１１１** | ０００１１１ | ０００００１１１ |
| ～ |  | ～ | ～ |
| ＋２９ |  | ０１１１０１ | ０００１１１０１ |
| ＋３０ |  | ０１１１１０ | ０００１１１１０ |
| ＋３１ |  | **０１１１１１** | ０００１１１１１ |
| ～ |  |  | ～ |
| ＋１２５ |  |  | ０１１１１１０１ |
| ＋１２６ |  |  | ０１１１１１１０ |
| ＋１２７ |  |  | **０１１１１１１１** |

**※ポイント**：最大値は、必ず先頭が[③　　　]で残りすべて[④　　　]である！

最小値は、必ず先頭が[⑤　　　]で残りすべてが[⑥　　　]である！

【５】シフト演算　　　　　　　　　　　　　　　　　　　　　　　　　　　　…テキストP.（　　　）

　1列に並んでいるビット列を、左または右方向に桁移動することをシフトという。**乗除算**を行う場合、コンピュータではシフトが用いられる。

　シフトの方法には、[①　　　　　　　　]と[②　　　　　　　　]がある。

１．論理シフト…符号ビットを無視して全ビットを左右にシフトさせる方法

[元となる数値]

①左へ２ビットシフト

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

②右へ２ビットシフト

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ０ | １ | ０ | ０ | ０ | ０ | １ | ０ |

２．算術シフト…符号ビットを考慮して符号ビット以外を左右にシフトさせる方法

[元となる数値：＋８]：正の場合

①４倍（２２）→左へ２ビット

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

　　※＋８×４＝３２

②１／４倍（２－２）→右へ２ビット

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

　　※＋８÷４＝２

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ０ | ０ | ０ | ０ | １ | ０ | ０ | ０ |

[元となる数値：－８]：負の場合

③４倍（２２）→左へ２ビット

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

　　※―８×４＝－３２

④１／４倍（２－２）→右へ２ビット

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

　　※―８÷４＝－２

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| １ | １ | １ | １ | １ | ０ | ０ | ０ |

≪範例１≫

数値を2進数で格納するレジスタがある。このレジスタに正の整数*x*を設定した後，“レジスタの値を2ビット左にシフトして，*x*を加える”操作を行うと，レジスタの値は*x*の何倍になるか。ここで，あふれ（オーバフロー）は，発生しないものとする。

ア　3 　　　　　イ　4 　　　　ウ　5 　　　エ　6

≪解答≫　ウ

シフトによるあふれが発生しないことを前提とした場合、左にｎビットシフトすると、元の値の２ｎ倍となります。

今回の問題では、*ｘ*を２ビット左にシフトした値と*ｘ*の加算であるため、

(*ｘ*×２２)＋*ｘ*＝(*ｘ*×４)＋(*ｘ*×１)＝*ｘ*×５

となります。

≪範例２≫

数値を２進数で表すレジスタがある。このレジスタに格納されている正の整数ｘを10倍にする方法はどれか。ここで，シフトによるあふれは，起こらないものとする。

ア　ｘを２ビット左にシフトした値にｘを加算し，更に１ビット左にシフトする。

イ　ｘを２ビット左にシフトした値にｘを加算し，更に２ビット左にシフトする。

ウ　ｘを３ビット左にシフトした値と，ｘを２ビット左にシフトした値を加算する。

エ　ｘを３ビット左にシフトした値にｘを加算し，更に１ビット左にシフトする。

≪解答≫　ア

ｘは正の整数であり、シフトによるあふれは起こらないので、ｘを１ビット、２ビット、３ビット左シフトすると、それぞれ2x、4x、8xとなります。

したがって、10xを求めるためには、xをまず２ビット左にシフトして4xを求め、これにxを足すことにより5xとし、さらに１ビット左にシフトすることによりその２倍、すなわち10xを求めます。