第２回
科学の甲子園 全国大会

実技競技 ③

君に届け！ 熱いメッセージ！

⌘ 実技マニュアル ⌘

本マニュアルは，科学の甲子園の実技競技を安全かつ公正に行うために各チームが守るべき事項などをまとめたものです。

実技競技は，全 47 チーム・150 人近くが参加し，体育館で行われます。学校の授業で行われる実験とは，環境や形態が異なります。したがって，各個人が十分に安全に配慮するとともに，まわりの様子にも気を配ることが必要です。

競技前のガイダンスをしっかり受けるとともに，本マニュアルを熟読してください。競技が，安全かつフェアに行われることを期待します。

実技競技を行うときには以下の項目を確認し，安全に行って下さい。

1. 実技にあたっての注意事項

会場の床には多くのケーブルが配線されている。安全を考えてテープで固定されているが歩くときにはつまずかないように足下に注意すること。

2. 課題の提出について

本実技では専用のソフトウエアを用いて課題プログラムを入力し実行する。公開コンテストの開始前に , 作成したプログラムを後述する「提出ボタン」を押して提出すること。なお, 公開コンテスト開始後はプログラムの変更はできない。

3. 機器 • 用品の確認

- 各チームのテーブルの上には今回の実技で必要な機器等がすべてセットアップされている。
  必要なものは以下の通りなので実技開始前に必ず確認すること。
- 不足している機器等がある場合は手を挙げて係員に申し出ること。なお，実技開始後，機器が故障した場合は交換するので，その場合も手を挙げて係員に申し出ること。

□ ノートパソコン
□ センサデバイス
□ ベースステーション
□ USB ケーブル (2 本)
□ 定規
□ 分度器
□ プログラム説明書記入用紙

4. 機器の取り扱い

センサデバイス

- 充電のために USB ケーブルに接続されているが, センサデバイスからの情報は無線で通信されるので, 競技中は自由に外して操作して構わない。本マニュアルの 5 に操作マニュアルを付けてあるので, 必要に応じて参照すること。

ノートパソコン

- 一般的なノートパソコンである。
- デスクトップにある PLMAM1Pad と書かれたアイコンをダブルクリックして, 本大会用の専用ソフトウエア PLMAM1Pad を起動して利用すること。

ベースステーション

- ノートパソコンがセンサデバイスと無線で通信するための機器である。特別な操作は必要ないが, 決して取り外さないこと。

5. センサデバイス操作マニュアル

センサデバイスは図 1 のような形状をしており，正面上部に LED 表示器が 8 つ並んでいる。また加速度センサを内蔵しており，直交する XYZ 軸それぞれの方向の加速度成分を短い時間間隔で測定することができる。さらに，その測定結果を無線でノートパソコンに逐一送信することができる。

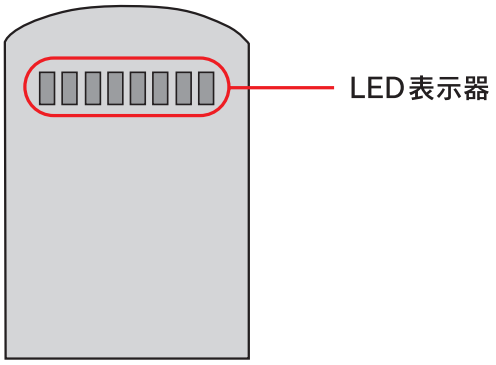

図1　センサデバイス

センサデバイスの下部には，図 2 のようにコントロールボタンとパワー LED がついている。センサデバイスの電源が切れているときには，コントロールボタンを押すと電源を入れることができる。また，電源が入っているときに，コントロールボタンを 5 秒以上押し続けると，電源を切ることができる。電源が入っているときに，コントロールボタンを短く押すと，デバイスをリセットすることができる。

パワー LED は，バッテリ残量を表示する。パワー LED が緑色に光っているときは，バッテリ残量が十分にある。パワー LED が赤色になったときは，バッテリ残量があまりない状態なので，USB ケーブルを接続して充電するか，センサデバイスの交換を申し出ること。

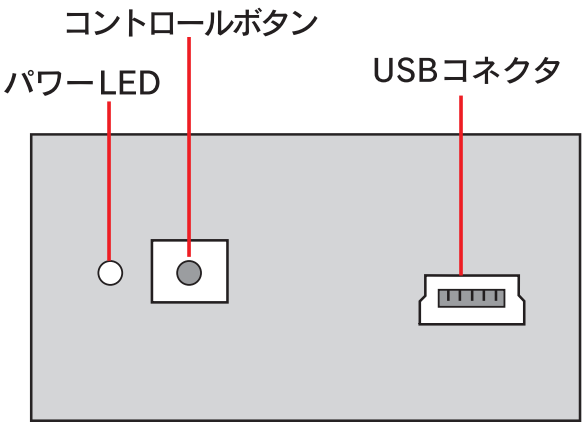

図 2　センサデバイスの下部

6. 競技用ソフトウエア操作マニュアル

6.1 競技用ソフトウエア操作画面

図 3 に競技用ソフトウエアの操作画面を示す。プログラム入力エリアに，6.2 で説明するプログラムを入力し，実行ボタンを押すと，（エラーがなければ）プログラムを実行することができる。プログラムの実行結果（文字列）は，メッセージ表示エリアに出力される。また，画面下部には，センサデバイスが検出した加速度の X,Y,Z 成分が，リアルタイムに表示される。

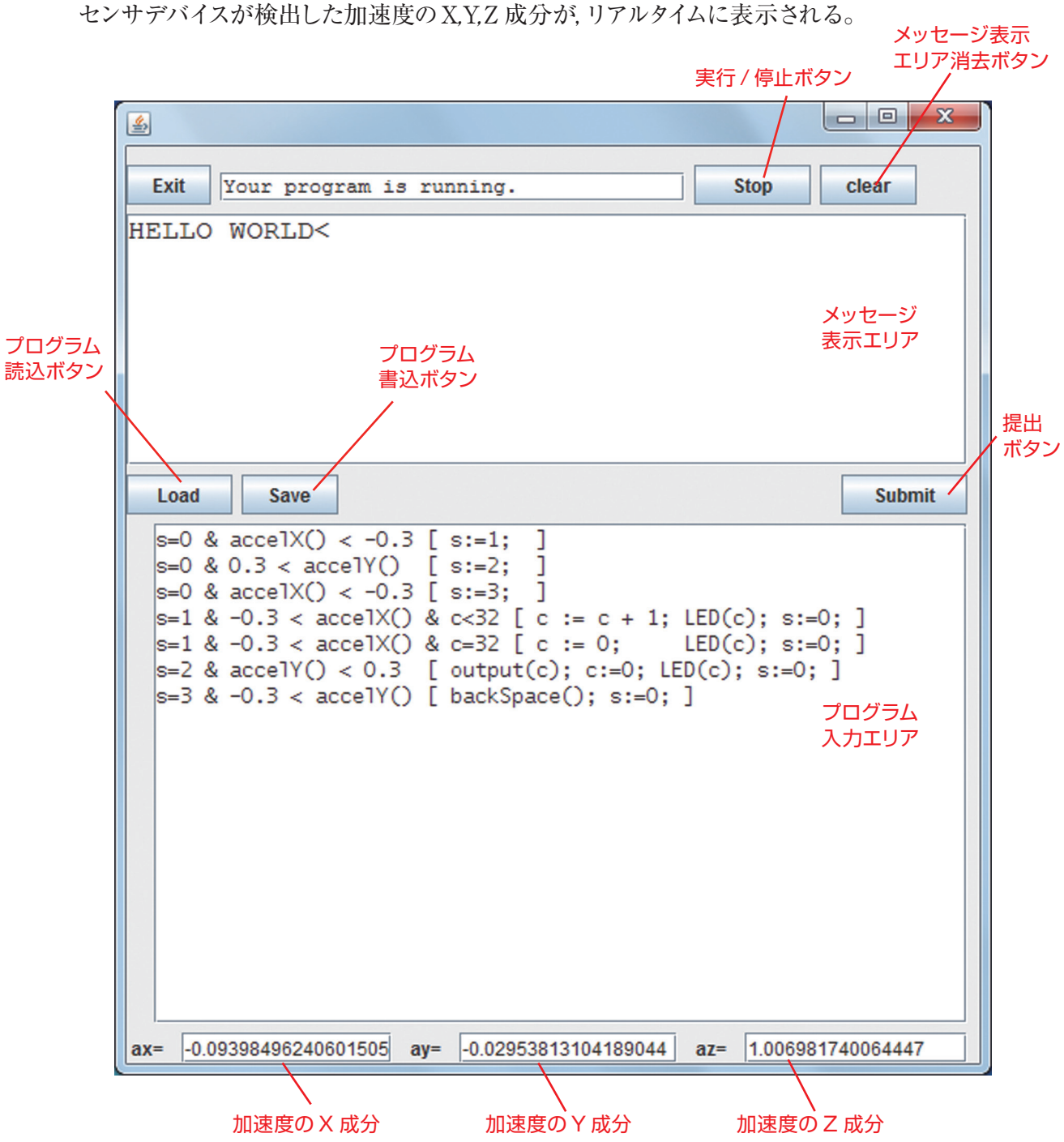

図 3

プログラムは，プログラム入力エリアに直接入力して作成してもよいが，メモ帳等のエディタを使って作成することもできる。「プログラム読込ボタン」を押すと，図４のようにファイル選択画面が表示されるので，メモ帳等で作成したプログラムを選択しファイルを開くと，プログラム入力エリアに選択したプログラムが取り込まれる。また，プログラム入力エリアでプログラムを作成したり変更したりした場合には，「プログラム書込ボタン」を押してファイルを選択し，プログラムを指定されたファイルに書き込むことができる。

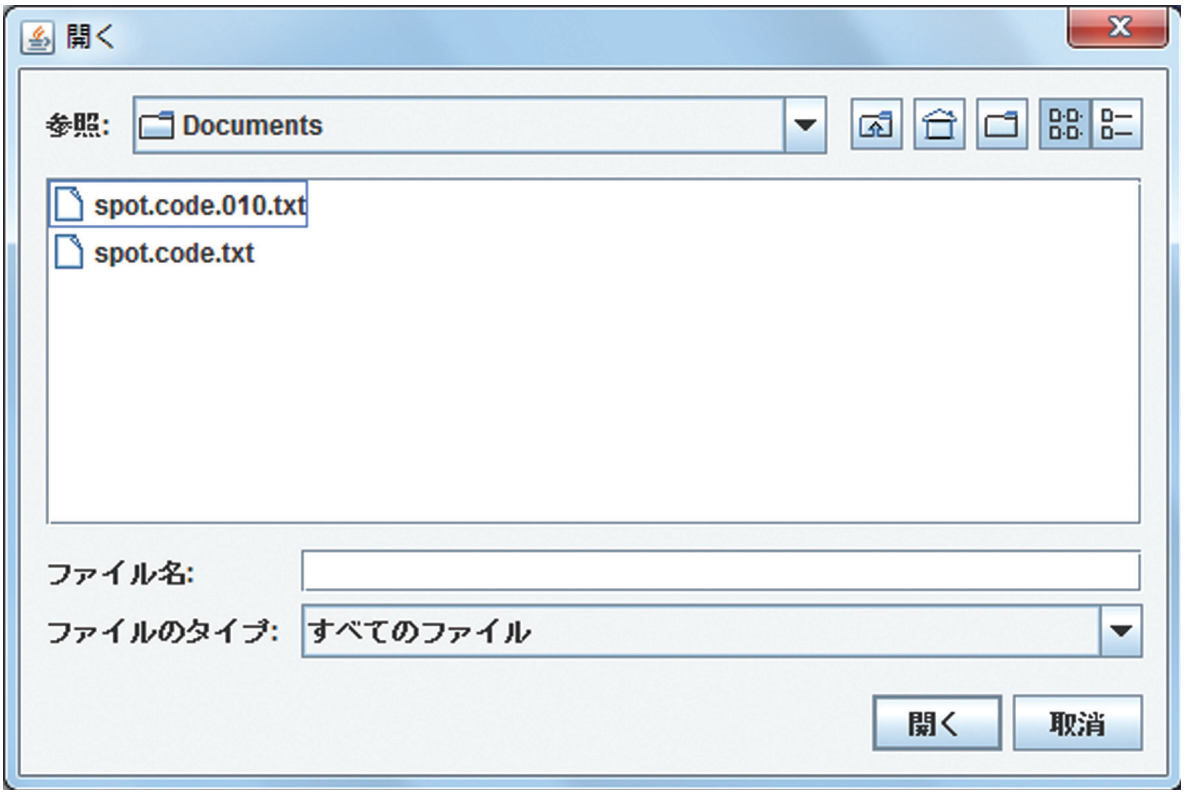

図４

プログラムの実行と停止は,「実行 / 停止ボタン」で行う。プログラムが動作していないときにこのボタンを押すと，プログラム入力エリアにあるプログラムが実行される。また，プログラムが動作しているときにこのボタンを押すと，実行中のプログラムが停止する。

プログラムの実行中にも，プログラム入力エリアのプログラムは書き換えることができる。ただし，いったん実行中のプログラムを停止させて，もう一度実行ボタンを押さないと，プログラムの変更は反映されない。

なお，課題プログラムの提出は指示のあったときに「提出ボタン（Submit）」を押して行うこと。

6.2 プログラミング言語 PLMAM1

本競技用ソフトウエアには，特別なプログラミング言語 PLMAM1（ピーエルマムワン）が組み込まれており，それを使用してプログラミングを行う。プログラムは，以下に示されているように，文がひとつ以上並べられたものとして構成されている。なおプログラムの途中にあるスペースや改行は，あってもなくてもプログラムの動作には関係ない。

文1　文2　文3　…　文n

各文は，条件式と処理の列から構成されている。条件式の記述が最初にあり，処理の列がそれに続く。処理の列は，角括弧［　］で囲まれている。

条件式［ 処理の列 ］

プログラムが実行されると，文 1 から順に各文の条件式の真偽値を調べ，最初に真となった文の処理の列を実行し，プログラムの最初に戻る。すべての条件式が偽であった場合も，プログラムの最初に戻る。以後，同じことを繰り返す。

以下，条件式と処理の列について説明する。

6.2.1 条件式

条件式は，式と論理演算子から構成されている。

式1　論理演算子1　式2　論理演算子2　…　論理演算子n　式n

ここで，論理演算子とは，&（論理積）または ｜（論理和）のいずれかを指す。また，式の中では，数値，変数，加減乗除，比較演算，関数呼び出しを扱うことができる。必要であれば，括弧 () で囲むことによって，演算を行う順序を変えることができる。

(a) 数値

整数または実数のことである。整数の値の範囲は，-2147483648 から 2147483647 まで扱える。実数は，扱える値の範囲がとても広いため，通常は値の範囲をあまり気にする必要はない。

例）　10　　整数の 10
　　—3.14　　実数の –3.14
　　—2.3E199　　実数の -2.3×10^{199}

(b) 変数

変数とは，整数または実数の値に，名前を付けて記憶したものである。変数には，(g) で説明する代入文により，特定の値を記憶させることができる。また，名前に基づいて記憶してある値を参照することもできる。変数に付ける名前は，アルファベットから始まる文字列であり，2 文字目以降はアルファベットまたは数字を自由に使うことができる。なお，大文字と小文字は区別される。また，全ての変数は，プログラムの実行開始時に 0 に初期化されている。

例） `a123` 変数 a123
`A123` 変数 A123
`ProgrammingLanguageForMakingAtsuiMessage1` 長い名前の変数

(c) 加減乗除

加減乗除は，＊，/，+，－ の各演算子により計算することができる。＊ と / は，+ と － よりも先に計算される。ただし，括弧で囲まれている場合には，その中が先に計算される。整数どうしの除算は，丸められて整数になる。除数か被除数のいずれかに実数が含まれている場合は，実数として計算される。

例） `(1+2)＊3` (1+2) × 3
`1+2＊3` 1+2 × 3
`10/3` 結果は 3
`10/3.0` 結果は 3.333 …

(d) 関数呼び出し

予め定義されている以下の関数を呼び出すことができる。

関数の名前	役割	値
`accelX(　)`	加速度の X 軸方向の成分を返す	加速度の X 軸方向の成分（実数）
`accelY(　)`	加速度の Y 軸方向の成分を返す	加速度の Y 軸方向の成分（実数）
`accelZ(　)`	加速度の Z 軸方向の成分を返す	加速度の Z 軸方向の成分（実数）

X, Y, Z それぞれの座標軸の向きは，図 5 に示された通りである。また，加速度の単位は G（約 9.8m/s^2）である。

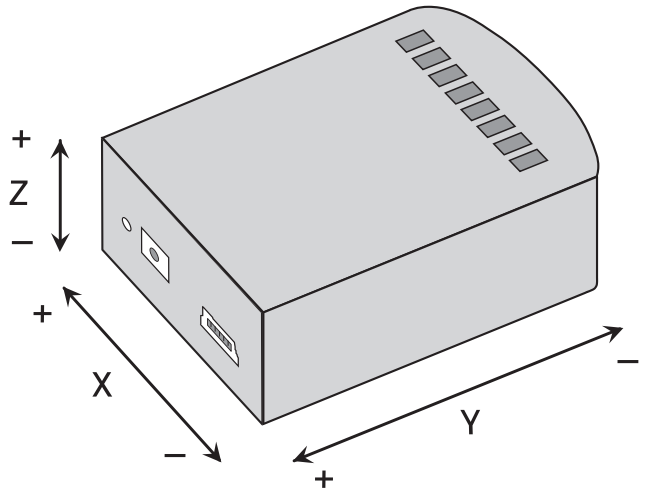

図5

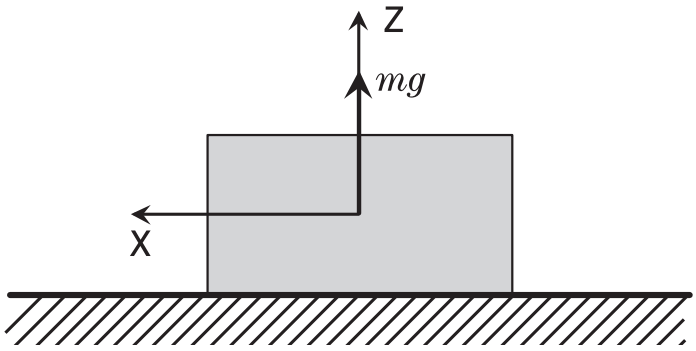

図6

ここで，センサデバイスが検出する加速度について，説明しておこう。センサデバイスは，外部から加わる力のうち重力を除いた力の大きさを，加速度として検出する。例えばセンサデバイスを，図 6 のように水平な机の上に置いて静止させると，センサデバイスに実際に加わっている力は , 重力に逆らってセンサデバイスを静止させている机からの垂直抗力のみである。垂直抗力の向きは Z 軸の正の方向であり，大きさは，加速度センサデバイスの質量 m 重力加速度の大きさを g とすると , mg である。このとき加速度センサは，Z 軸の正の向きに大きさ 1.0 の加速度を検出する。すなわち, `accelX ()` と `accelY ()` は 0.0 となり, `accelZ ()` の値は 1.0 になる (実際の測定値には , 測定誤差やノイズが含まれている)。

つぎに Y 軸の正の方向を手前にして，図 7 のようにセンサデバイスを反時計回りに 30 ˚傾けて静止させたとすると, 手が支えている力の向きは図 7 のようになり, 大きさは mg となる。各座標軸に, この力を分解すると X 軸方向に $-mg\sin 30^\circ$，Z 軸方向に $mg\cos 30^\circ$，Y 軸方向には 0 になる。この状態において各関数の返す値は，それぞれ `accelX ()` =−0.5 (= $-1/2$), `accelY ()` = 0.0, `accelZ ()` = +0.8660 (= $+\frac{\sqrt{3}}{2}$) である。

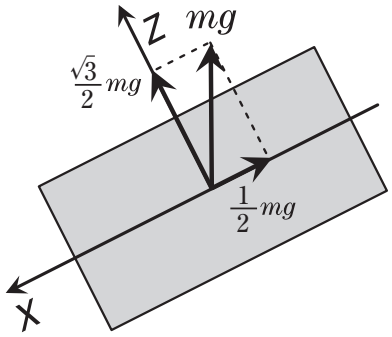

図7

さらにセンサデバイスを手で持って動かしたとすると，センサデバイスは，重力に逆らって支える力を含め，手から受ける加速度を検出する。例えば，図 8 のようにセンサデバイスを手で持ち，水平に保ったまま X 軸の正の方向に，手からセンサデバイスに力 $F\ (=ma)$ を加えて動かしたとする。このとき `accelX ()` の値は $\frac{a}{g}$ になる。Y 軸方向には力を加えていないので，`accelY ()` の値は0.0である。また手はセンサデバイスをZ軸の正の方向に重力に逆らう力を加えて支えているので，`accelZ ()` の値は 1.0 になる。

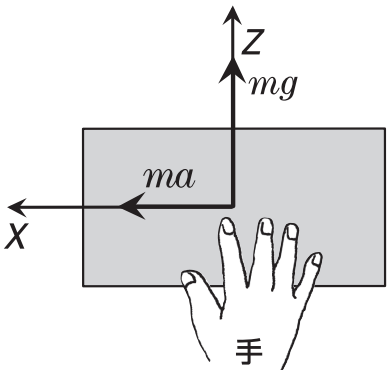

図8

(e) 比較演算子

数の大小比較を行って，真偽値を返す。= (等しい), > (大きい), < (小さい) の三つの演算子がある。これらの演算子の優先順位は，加減乗除よりも低い。

例） `a > 100`　　変数 a の値が 100 より大きいときに真，それ以外は偽
`x = 2 * b`　　変数 x の値と 2 * b の値が等しいときに真，それ以外は偽

(f) 論理演算子

論理演算子は真偽値を受け取り, 論理積(&) もしくは論理和(|) を計算して真偽値を返す。論理演算子の優先順位は，比較演算子よりも低く，他の演算子よりも後に計算される。

例）`a > 100 & b = 100`　変数 a の値が 100 より大きく，かつ変数 b の値が 100 と等しいときに真，それ以外は偽

6.2.2 処理の列

処理の列とは，以下のような形式で，セミコロン (;) を末尾につけた処理をゼロ個以上並べたものである。全体は，角括弧 ([と]) でくくって，条件式の後に記述する。

処理 1; 処理 2; 処理 3; … ; 処理 n;

各々の処理は，代入文もしくは手続き呼び出しのいずれかである。以下，順に説明する。

(g) 代入文

(b) で説明した変数に対し値を割り付けて記憶する処理であり , 以下のように記述する。

変数名 := 式

代入文を実行すると，まず式の値を計算し，次にその計算結果を変数に記憶する。式の値は，整数か実数でなくてはならない。式は,「6.2.1 条件式」の項目で説明した (a) から (d) の各要素から構成される。

例） `ax2 := accelX () * accelX ()`　加速度の X 成分の 2 乗を計算して，結果を変数 ax2 に格納する。

(h) 手続き呼び出し

ここで記述できる手続き呼び出しは，次のいずれかである。

手続きの名前	役　割
output（文字コード）	与えられた文字コードに対応する文字を，文字表示領域に追加する。
backSpace()	最後に文字表示領域に追加された一文字を削除する。
LED（数値）	センサデバイスの LED に，与えられた数値を8桁の2進数で表示する。0 は暗い赤，1 はそれ以外の色で表示される。

output 手続きで指定できる文字コードは 0 から 31 の 32 種類で，それぞれ以下の表に示された文字に対応している。

コード（二進表示）	文字	コード（二進表示）	文字	コード（二進表示）	文字
0 (0000000)	空白	11 (0001011)	K	22 (0010110)	V
1 (0000001)	A	12 (0001100)	L	23 (0010111)	W
2 (0000010)	B	13 (0001101)	M	24 (0011000)	X
3 (0000011)	C	14 (0001110)	N	25 (0011001)	Y
4 (0000100)	D	15 (0001111)	O	26 (0011010)	Z
5 (0000101)	E	16 (0010000)	P	27 (0011011)	,（カンマ）
6 (0000110)	F	17 (0010001)	Q	28 (0011100)	.（ピリオド）
7 (0000111)	G	18 (0010010)	R	29 (0011101)	!
8 (0001000)	H	19 (0010011)	S	30 (0011110)	?
9 (0001001)	I	20 (0010100)	T	31 (0011111)	'（アポストロフィ）
10 (0001010)	J	21 (0010101)	U		

6.2.3 プログラムの例

LED ランプが上面, Y 軸の正の向きが手前に来るようにセンサデバイスを持った上で, センサデバイスを前後左右に傾けることによって文字入力を行う方法を考えてみよう (図 9)。例えば, 以下のルールに従って, デバイスを傾ける動作と文字入力とが関係づけられているものとする。

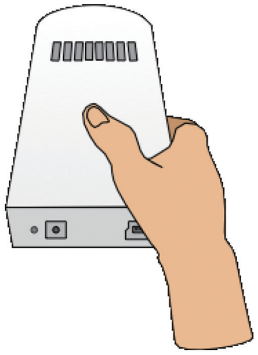

図9

(1) デバイスをほぼ水平に置いた状態(accelX() と accelY() がほぼゼロ)を基準にする。
(2) デバイスを左側を押し下げる (accelX () がマイナス) と, 文字コードを表す変数 c を1増やす。
(3) デバイスの先を押し下げる (accelY () がプラス) と , 変数 c に対応する文字を出力する。
(4) デバイスの先を引き上げる (accelY () がマイナス) と, 表示エリアから最後に出力された文字を一文字削除する。

以上の方針をもとに実際のプログラムを記述したものが図 10 である。各行の先頭に説明用の行番号を付けてあるが, これらは実際には記述しない。

```
1: s=0 & accelX() < -0.4 [ s:=1; ]
2: s=0 & 0.4 < accelY()  [ s:=2; ]
3: s=0 & accelY() < -0.4 [ s:=3; ]
4: s=1 & -0.2 < accelX() & c<32 [c := c + 1; LED(c); s:=0; ]
5: s=1 & -0.2 < accelX() & c=32 [c := 0;      LED(c); s:=0; ]
6: s=2 & accelY() < 0.2  [ output(c); c:=0; LED(c); s:=0; ]
7: s=3 & -0.2 < accelY() [ backSpace(); s:=0; ]
```

図10

図 10 のプログラムの動作を詳しく見てみることにしよう。まず, プログラムの実行開始時には, s=0 となっていることに注意しよう。次に, センサデバイスがほぼ水平に置かれている場合を考えてみると, 1 ～ 7 行目のいずれの条件にも当てはまらないので , 何の処理も実行されないことが分かる。この状態から, センサデバイスの左側を押し下げる, 先頭を押し下げる, 先頭を引き上げる,

といった動作を行うと，それぞれ 1 行目，2 行目，3 行目の条件式が真になる。この結果，変数 s に 1,2,3 がそれぞれ代入される。

s=1 でかつセンサデバイスの左側が押し下げられた状態では,どの条件にも当てはまらないので，特に何も処理は実行されない。この状態からセンサデバイスを水平に近い位置まで戻すと，変数 c の初期値は 0 なので，4 行目の条件式が真になる。その場合の処理内容は，変数 c に 1 加算し（c:=c+1;），c の値を LED に表示し（LED（c）;），s=0 の状態に戻す（s:=0;）というものである。変数 c の値が大きくなり 32 に達すると，4 行目の条件式が真になる代わりに，5 行目の条件式が真になり，c が 0 に戻される（c:=0;）。

s=2 でかつセンサデバイスの先頭が押し下げられた状態では，どの条件にも当てはまらないので，特に何も処理は実行されない。この状態からセンサデバイスを水平に近い位置まで戻すと，6 行目の条件式が真になる。その場合の処理内容は，文字コード c の文字を出力し，変数 c の値を 0 に戻し,c の値を LED に表示し（LED（c）;),s=0 の状態に戻す（s:=0;）というものである。

s=3 でかつセンサデバイスの先頭が引き上げられた状態では，どの条件にも当てはまらないので，特に何も処理は実行されない。この状態からセンサデバイスを水平に近い位置まで戻すと，7 行目の条件式が真になる。その場合の処理内容は，最後に入力された文字を削除し（backSpace（）;），s=0 の状態に戻す（s:=0;）というものである。

以上で見てきた通り，変数 c に 0 から 31 までの任意の値を設定できるので，手間はかかるが任意の文字列を出力できることが分かる。

なお，傾けたときの加速度の成分の絶対値が 0.4 以上であることと，水平にもどしたときの加速度の成分の絶対値が 0.2 以下であるのは，人が手で操作するときの不安定さに対する工夫である。