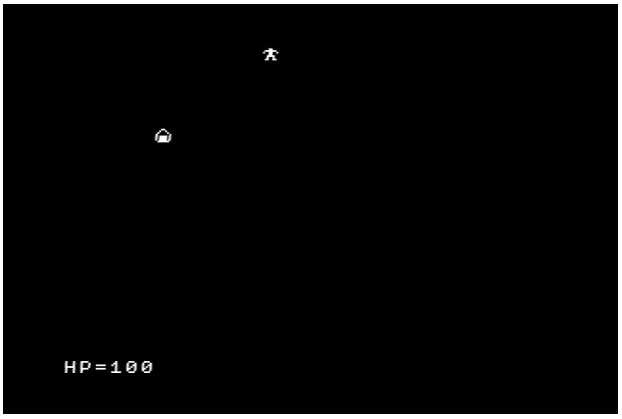
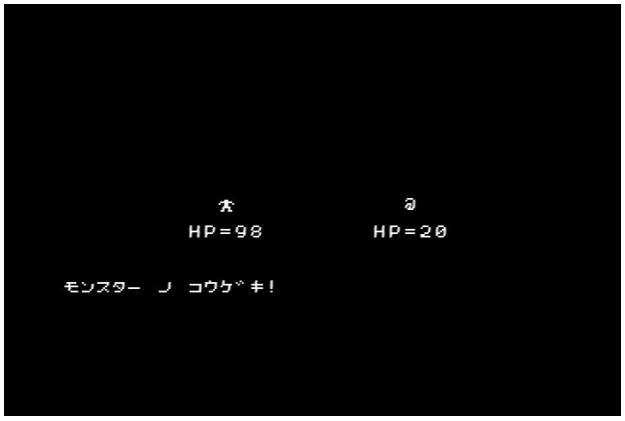
【A-2】IchigoJam でロールプレイングゲームを作る

●今回の目標

IchigoJam で動くロールプレイングゲーム(RPG)を作ります。
自分のキャラクター(勇者)がマップの中を移動して、モンスターと出会ったら戦うゲームです。





●マップ画面を表示する

まず、プログラムの初期設定をした後、画面に HP(勇者の体力)を表示します。

```
10 '*RPG
20 CLV
30 H=100
300 '*MAP
310 CLS
320 LOCATE 0,23
330 PRINT "HP=";H;
```

- コメントで、プログラムのタイトルを入れる
- 変数をクリア
- 変数 H を 100 にする
- コメントを入れる
- 画面をクリアする
- カーソルを画面左下へ移動
- HP を表示

入力できたら、「RUN」で実行してみましょう。
画面がクリアされて、左下に HP が表示されます。
続いて「OK」が表示されますが、今回は気にしないことにします。

プログラムの内容を説明します。

10 行:コメントで、プログラムのタイトルを入れています。
先頭に「’ 」(アポストロフィ、キーボードでは Shift キーを押しながら「7」を押す)を付けると、その行はコメントとなり、何も実行されません。ですから、「’ 」のあとは好きな文字を書くことができます。
プログラムを後で見た時にわかりやすくするために、プログラムにいろいろコメントを入れるといいでしょう。

20 行:CLV(シーエルブイ)命令は、**変数**(へんすう)をクリアする命令です。
変数とは、数字を入れる入れ物(箱)と思ってください。
小学校の算数でやる「□」(四角)、中学校の数学でやる「x」と考えればいいです。
今回は、ゲームの最初に、全ての変数をクリアしています。

30 行:勇者の体力を表す変数 H に 100 を入れています。
「H＝１００」は、「H と 100 が等しい」という意味ではなく、「H に 100 を入れる」(代入する)という意味です。

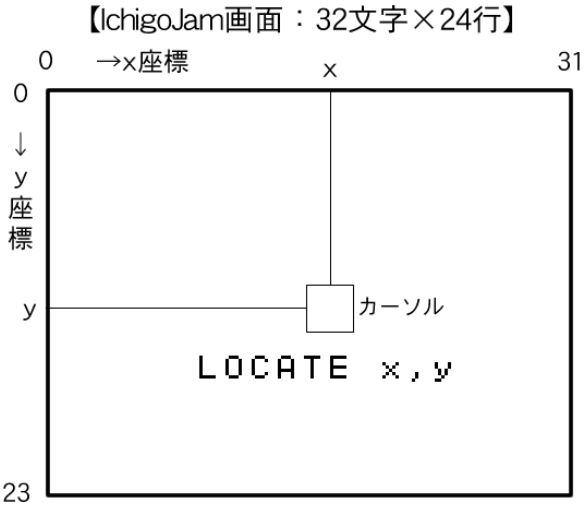
300 行:「ここからマップのプログラム」と示すコメントを入れています。
※あとのプログラムのために、行番号を 300 にしています。

310 行:LOCATE(ローケート)命令は、画面に文字を表示する位置(カーソル位置)を設定する命令です。

```
LOCATE  0  ,23
        x 座標  y 座標
```

x 座標	画面の x 座標(0～31)。
y 座標	画面の y 座標(0～23)。

IchigoJam の画面サイズは、横 32 文字×縦 24 行になっています。
横(x 方向)の座標は 0～31、縦(y 方向)の座標は 0～23 になっています。
今回は「LOCATE 0,23」として、画面左下の座標を指定しています。



320 行:「PRINT "HP=";H;」で、画面に変数 H の値を表示しています。

★LOCATE 命令の座標の数字を変えると、HP の表示位置が変わります。
いろいろ変えて試してみましょう。

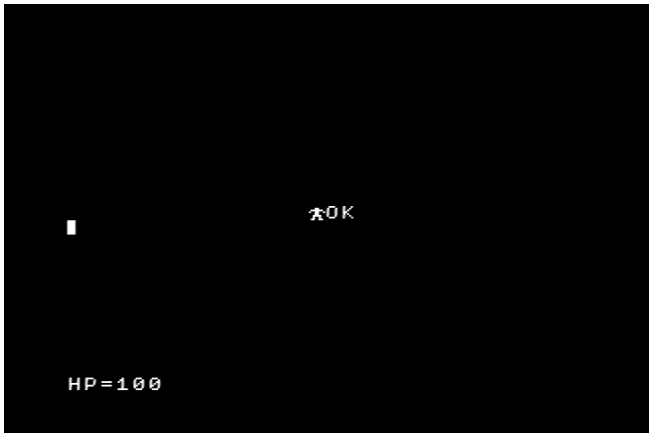
●勇者を表示する

マップの上に、勇者（自分）を表示してみましょう。
以下のプログラムを追加します。

```
10 '*RPG
20 CLV
30 H=100
40 X=15:Y=11
50 GOSUB 300
290 END
300 '*MAP
310 CLS
320 LOCATE 0,23
330 PRINT "HP=";H;
340 LOCATE X,Y
350 PRINT CHR$(250);
390 RETURN
```

- 40：勇者の x 座標、y 座標を設定
- 50：マップを表示するサブルーチンを呼ぶ
- 290：プログラムを終了する
- 340：カーソルを座標(X,Y)へ移動
- 350：勇者を表示
- 390：メインプログラムへもどる

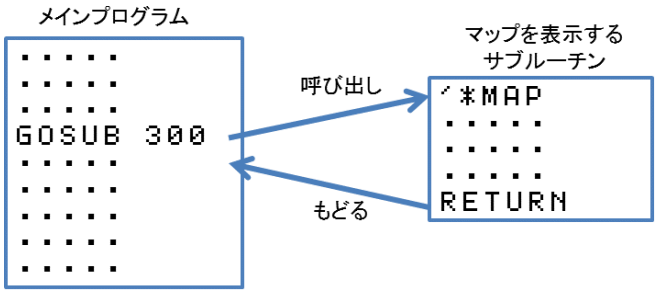
プログラムを実行してみましょう。
画面中央に勇者が表示されます。



追加したプログラムの内容を説明します。

40 行：300 行からのマップ表示のプログラムをサブルーチンとして、GOSUB（ゴーサブ）命令で呼び出します。

メインプログラムからは「GOSUB」命令でサブルーチンへジャンプし、サブルーチンからは「RETURN」（リターン）命令でもどります。もどった後は、GOSUB 命令の続きへプログラムの処理が移ります。



マップを表示する処理をあとでまた使うため、このようにサブルーチンにします。

50 行：勇者の横座標の変数 X、縦座標の変数 Y に値を入れています。

★X、Y の数字を変えると、勇者の表示位置が変わります。
いろいろ変えて試してみましょう。

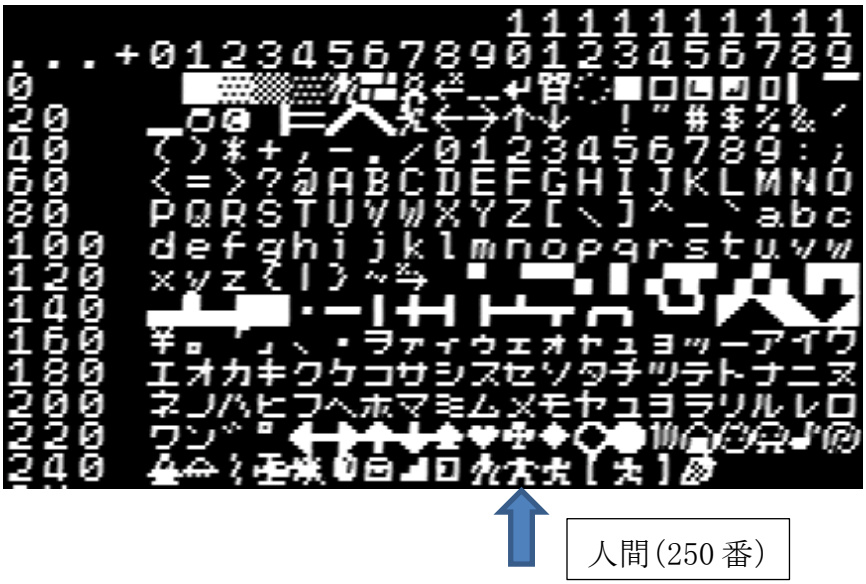
60 行：LOCATE 命令で、カーソルを(X,Y)の座標へ移動します。
このように、座標を変数で指定することもできます。

70 行：勇者のキャラクターを画面に表示します。ここでは勇者の文字を **CHR$**（シー・エッチ・アール・ドル、キャラクター）関数で指定しています。

```
CHR$( 250)
```
文字コード

文字コード	0～255 の数字で文字コードを指定

今回は文字コード 250 番を指定して、人間を画面に表示しています。
IchigoJam の文字コードは、右のようになっています。



★CHR$関数の文字コードを変えると、勇者のキャラクターが変わります。試してみましょう。

290 行：**END**（エンド）命令で、プログラムを終了します。
ここで終了しておかないと、そのまま次のマップ表示サブルーチンを実行してしまいます。

390 行：**RETURN** 命令で、メインプログラムへもどります。

●勇者を動かす

勇者をカーソルキー（矢印キー）で上下左右に動かせるようにしてみましょう。
以下のプログラムを追加します。

```
…（前略）…
40 X=15:Y=11
50 GOSUB 300
100 '*GAMELOOP
110 U=X:V=Y
120 IF BTN(LEFT)=1 THEN U=U-1
130 IF BTN(RIGHT)=1 THEN U=U+1
140 IF BTN(UP)=1 THEN V=V-1
150 IF BTN(DOWN)=1 THEN V=V+1
200 LOCATE X,Y
210 PRINT " ";
220 LOCATE U,V
230 PRINT CHR$(250);
240 X=U:Y=V
250 WAIT 5
260 GOTO 100
290 END
…（後略）…
```

コメント

次の勇者の表示座標 U,V を設定

←キーが押されたらＵを１減らす。
→キーが押されたらＵを１増やす。

↑キーが押されたらＶを１減らす。
↓キーが押されたらＶを１増やす。

勇者を消す

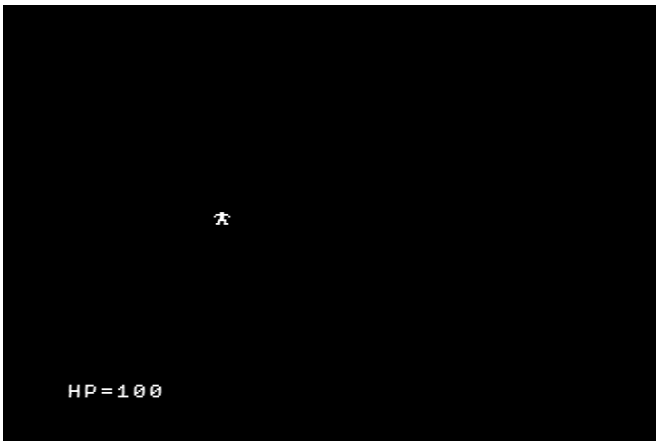
次の位置(U,V)に勇者を表示

X,Y を次の位置に更新

時間待ち

ゲームループの最初へもどる

プログラムを実行してみましょう。
カーソルキーの上下左右を押すと、勇者が移動します。
このままだとプログラムがいつまでも終わらないので、ESC キーを押して止めてください。



追加したプログラムの内容を説明します。

100 行：ここからゲームの処理をするプログラムになるので、コメントを入れています。
ループしてくり返すので、「GAMELOOP」としています。

110 行：勇者の次の位置の座標(U,V)を、今の位置(X,Y)と同じに設定します。

120 行:勇者を左へ動かす処理をします。

BTN(ボタン)関数を使って、左矢印キーが押されているかどうかを判断します。
BTN 関数の文法は以下のとおりです。

```
BTN( LEFT)
     キー指定
```

キー指定	LEFT…左矢印キー(←) RIGHT…右矢印キー(→) UP…上矢印キー(↑) DOWN…下矢印キー(↓) SPACE…スペースキー
返り値	キーが押されている＝1 キーが押されていない＝0

指定したキーが押されているとBTN 関数の値が 1、押されていないと 0 になります。
その BTN 関数の値を、**IF**(イフ)命令で判断します。

```
IF BTN(LEFT)=1 THEN U=U-1 ELSE ~
   条件式          条件が成り立つ時に実行   条件が成り立たない時に実行
```

条件式	条件を判断する式。
THEN(ゼン)の後	条件が成り立つ時に実行するプログラム。
ELSE(エルス)の後	条件が成り立たない時に実行するプログラム。 ELSE 以下は省略可能。

この行は、「もし BTN(LEFT)が 1 だったら(＝左矢印キーが押されていたら)、変数 U を 1 減らす」という意味になります。
なお「U=U-1」は、「**U から 1 を引いて、それを U に入れる**」という意味です。「U と U-1 が等しい」という意味ではないので注意してください。

130～150 行:120 行と同じように、勇者を右・上・下へ動かす処理をします。

200～210 行:元の位置(X,Y)の勇者を消します。

220～230 行:次の位置(U,V)に勇者を表示します。

240 行:勇者の位置(X,Y)を、次の位置(U,V)に更新します。

250 行:勇者の動きを少し遅くするため、**WAIT**(ウェイト)命令で時間待ちをします。

260 行:続けて勇者を動かすため、GOTO(ゴートゥー)命令で前へ戻しています。

```
GOTO 100
     行番号
```

指定した行番号へ実行を移します。

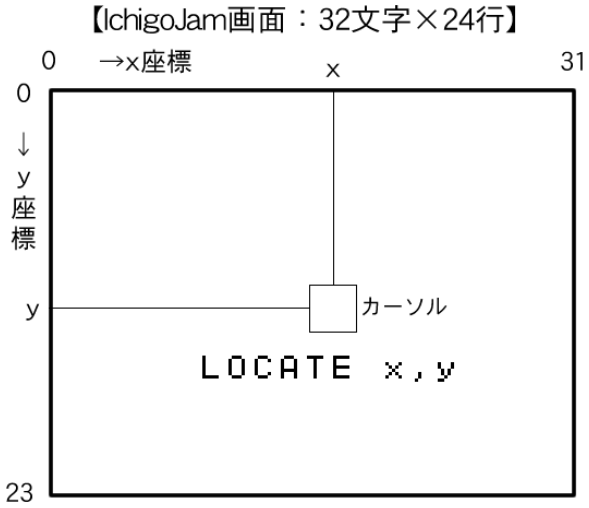
このプログラムだと、以下の問題があります。

- 勇者が画面上はじに行っても上矢印キー「↑」を押し続けると、勇者が消えてしまう。
- 画面左はじ、右はじ、下はじでもおかしな動きをする

これは、勇者の座標(X,Y)の範囲を考えずに増やしたり減らしたりしてしまい、X や Y がマイナスの値になってしまったりするからです。

画面サイズから考えて、勇者を動かす範囲を以下のようにします。

x 座標:0～31
y 座標:0～22
※y=23 の場所には「HP=100」の表示があるため、23 にはならないようにする。



勇者を動かす範囲を設定して、プログラムを改造します。

```
…(前略)…
120 IF BTN(LEFT)=1 AND U>0 THEN U=U-1
130 IF BTN(RIGHT)=1 AND U<31 THEN U=U+1
140 IF BTN(UP)=1 AND V>0 THEN V=V-1
150 IF BTN(DOWN)=1 AND V<22 THEN V=V+1
…(後略)…
```

←キーが押された、かつ、U が 0 より大きかったら、U を 1 減らす。以下も同様。

プログラムを実行してみましょう。勇者が画面をはみ出さなくなります。

今回は、IF 命令の条件式に、「**AND**」(アンド)でつないで、2つの条件を入れています。

```
120 IF BTN(LEFT)=1 AND U>0 THEN U=U-1
```

この条件式は、「BTN(LEFT)が 1 と等しい、**かつ**、U が 0 より大きい」という意味になります。
両方の条件が成り立った時だけ、「THEN」以下の命令が実行されます。
画面の左はじに行くと、U が 0 になるので、条件が成り立たず、それ以上は左へ行きません。
130～150 行も同様に、右・上・下はじへ行くと、それ以上は勇者が進まないようにしています。

「SAVE 0」でプログラムを保存しておきましょう。

●村人を表示する

わき役として村人を登場させて、勇者が村人と出会ったら話をするようにしてみましょう。
まず、プログラムの最初で、村人の位置を設定します。

```
10 '*RPG
20 CLV
30 H=100
40 X=15:Y=11
42 P=RND(32):Q=RND(23)
50 GOSUB 300
```

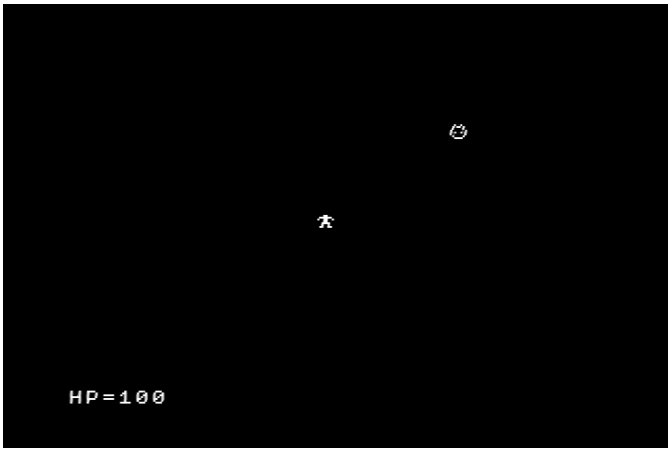
村人の座標(P,Q)を乱数で設定

マップを表示するサブルーチンを改造して、村人を表示するようにします。

```
300 '*MAP
310 CLS
320 LOCATE 0,23
330 PRINT "HP=";H;
340 LOCATE X,Y
350 PRINT CHR$(250);
360 LOCATE P,Q
365 PRINT CHR$(236);
390 RETURN
```

座標(P,Q)に村人を表示

プログラムを実行してみましょう。
村人（ネコのキャラクター）が画面に表示されます。
プログラムを実行するたびに、村人の表示位置が変わります。



プログラムの内容を説明します。

42 行：村人の横座標 P・縦座標 Q を指定します。毎回同じ場所に出てくると面白くないので、**乱数**（らんすう）で指定します。乱数とは、毎回違うでたらめな数字です。
乱数を取り出すには、**RND**（ランダム）関数を使います。

```
RND(    32)
```
乱数の最大値

乱数の最大値	0～最大値-1 の乱数が出てきます。

「RND(32)」と指定すると、0～31 の乱数が出てくるので、村人の横座標 P が画面の左はじから右はじのどこかになります。
同様に縦座標 Q も、乱数で 0～22 のどこかにします。

360～365 行：村人を画面に表示します。文字を **CHR$**関数で指定しています。

★365 行の文字コードを変えれば、キャラクターが変わります。いろいろ試してみましょう。

●村人と話す

勇者が村人と出会ったら、話をするようにしてみましょう。
ゲームループの中で、勇者が移動した時に、村人と出会ったかどうかを判断します。

```
100 '*GAMELOOP
110 U=X:V=Y
120 IF BTN(LEFT)=1 THEN U=U-1
130 IF BTN(RIGHT)=1 THEN U=U+1
140 IF BTN(UP)=1 THEN V=V-1
150 IF BTN(DOWN)=1 THEN V=V+1
160 C=SCR(U,V)
170 IF C=236 THEN GOSUB 400
200 LOCATE X,Y
210 PRINT " ";
```

160 C=SCR(U,V) ：次の位置(U,V)にある文字を読み取る

170 IF C=236 THEN GOSUB 400 ：もし次の位置に村人がいたら 村人と話すサブルーチンを呼ぶ

400 行から、村人と話すサブルーチンを追加します。

```
400 '*PEOPLE
410 LOCATE 12,23
420 PRINT "コンニチハ";
430 WAIT 120
440 P=RND(32):Q=RND(23)
450 GOSUB 300
460 RETURN
```

400 '*PEOPLE ：コメント

410 LOCATE 12,23 / 420 PRINT "コンニチハ"; ：画面下に「コンニチハ」と表示

430 WAIT 120 ：時間待ち

440 P=RND(32):Q=RND(23) ：村人の座標(P,Q)を再設定

450 GOSUB 300 ：マップ表示サブルーチンを呼ぶ →村人が再表示される

460 RETURN ：もどる

プログラムを実行してみましょう。
勇者が村人に出会うと、画面の下に「コンニチハ」と表示されます。
その後、村人が違う位置へ移動します。

それでは、プログラムの内容を説明します。

160 行：SCR（スクリーン）関数で、勇者の移動先にある文字を読み取っています。

```
SCR(  U,  V)
```
x 座標　y 座標

x 座標	読み取りたい画面の x 座標（0～31）
y 座標	読み取りたい画面の y 座標（0～23）

関数の返り値	指定した座標に表示されている文字コード（0～255）

170 行：読み取った文字コードが 236（村人）だったら、村人と話すサブルーチンを呼びます。

410～420 行：画面下に、村人のセリフ「コンニチハ」を表示します。
★ここの文字を変えると、村人のセリフが変わります。いろいろ試してみましょう。

430 行：プレイヤーにセリフを読んでもらうために、WAIT 命令で時間待ちをします。

440 行：村人の次の位置を乱数で設定します。

450 行：マップ表示のサブルーチンを呼んで、村人を再度表示します。

460 行：ゲームループにもどります。

「SAVE 0」でプログラムを保存しておきましょう。

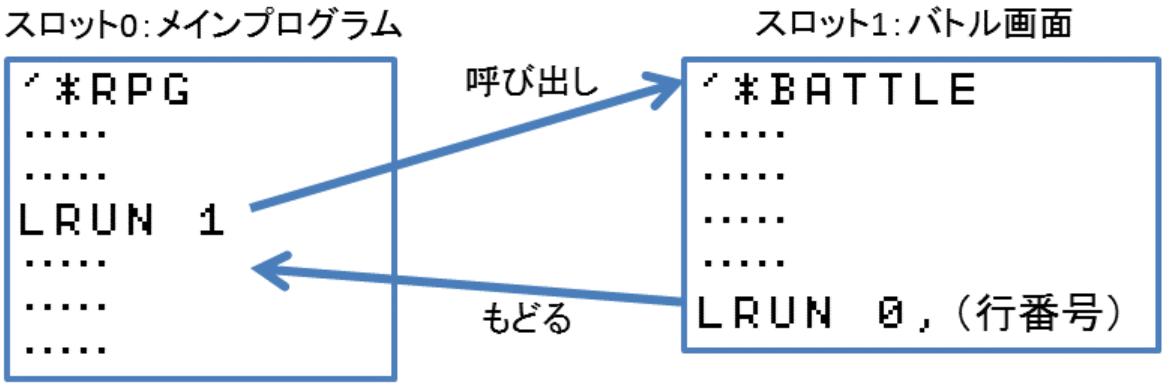
●バトル画面を作る

いよいよモンスターとバトルする画面を作ってみましょう。
ただし、ここまでのプログラムで IchigoJam の残り容量がかなり少なくなっているので、このままだとバトル画面が作れません。

※プログラムの残り容量は、FREE（フリー）関数で調べることができます。

IchigoJam のプログラム容量は、1 プログラム当たり 1024 バイトです。
この場合、残り容量が 330 バイトなので、全体の 2/3 ほど使っていることになります。

これを解決するために、プログラムスロットを 2 つ使って、2 本のプログラムを連動させるようにします。



これまでのプログラムは「SAVE 0」として、スロット 0 に保存します。
これから作るバトル画面のプログラムをスロット1に保存して、バトルする場面ではメインプログラムからスロット1のプログラムを呼び出します。
バトルが終わったら、またスロット 0 のメインプログラムへ戻ってくるようにします。

以下の作業を順番どおりにやります。

まちがえると、これまで作ったプログラムが消えてしまう恐れがあります。

注意して作業してください。

（1）今作っているメインプログラムを、**スロット 0** に保存します。

```
SAVE 0
Saved 000byte
OK
```

（2）「NEW」（ニュー）コマンドで、作業しているプログラムをクリアします。

```
NEW
OK
```

※（1）の保存を忘れると、ここでこれまで作ったプログラムが消えてしまいます。

（3）新しくバトル画面のプログラムを作ります。

次ページから、作っていきます。

（4）バトル画面のプログラムを作ったら、**スロット1** に保存します。

```
SAVE 1
Saved 000byte
OK
```

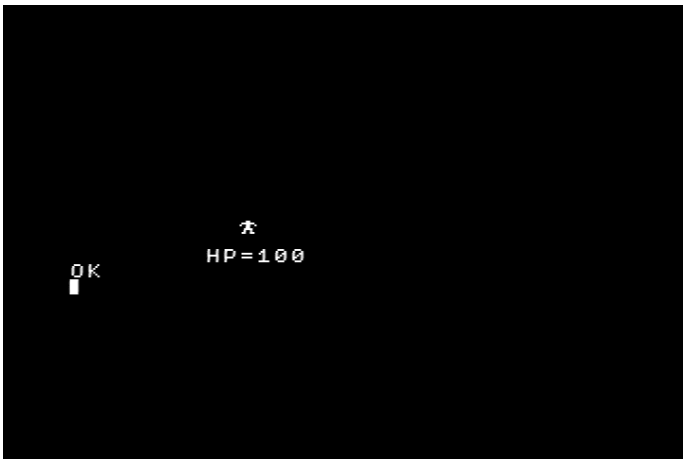
※まちがえてスロット 0 に保存すると、メインプログラムが消えてしまいます。

それでは、バトル画面のプログラムを作っていきます。
まず、画面をクリアして、勇者（自分）のキャラクターとHPを表示しましょう。

```
10 '*RPG-BATTLE
20 CLS:H=100
30 LOCATE 10,11
40 PRINT CHR$(250)
50 LOCATE 8,13
60 PRINT "HP=";H
```

コメントでタイトルを入れる
画面をクリア、勇者のHPを100にする
勇者のキャラを表示
勇者のHPを表示

プログラムを実行してみましょう。
画面左側に、勇者のキャラとHPが表示されます。



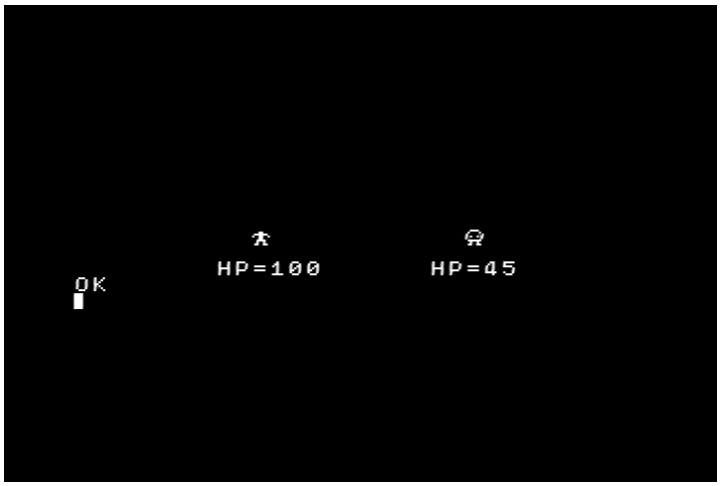
※20行で勇者のHP（変数H）を100に設定しています。
最終的にはメインプログラムから呼び出された時のHの値を使うので、
「**:H=100**」はいずれ消します。

次に、相手のモンスターを表示しましょう。

```
10 '*RPG-BATTLE
20 CLS:H=100
30 LOCATE 10,11
40 PRINT CHR$(250)
50 LOCATE 8,13
60 PRINT "HP=";H
70 LET [0],237,30,239,40,240,50
80 M=RND(3)*2
90 I=[M+1]+RND(21)
100 LOCATE 22,11
110 PRINT CHR$([M])
120 LOCATE 20,13
130 PRINT "HP=";I
```

配列変数に3種類のモンスターのキャラクターとHPを設定
モンスターの番号（0,2,4）を乱数で設定
モンスターのHPを配列の値＋乱数で設定
モンスターのキャラを表示
モンスターのHPを表示

プログラムを実行してみましょう。
画面右側にモンスターとHPが表示されます。
何度も実行すると、そのたびにモンスターの種類が変わります。



今回は、モンスターの種類を変えるのに、**配列変数**（はいれつへんすう）を使っています。

```
70 LET [0],237,30,239,40,240,50
```

配列変数は、番号がついている変数です。
[0]、**[1]**、**[2]**…と、中かっこと数字であらわします。
IchigoJamでは、[0]～[101]の102個の配列変数が使えます。

```
LET [0],237,30,239,40,240,50
```

このようにLET（レット）命令を使うと、配列変数[0]～[5]に、6個の数字が入れられます。

```
[0]=237:[1]=30:[2]=239:[3]=40:[4]=240
:[5]=50
```

と同じです。図で書くと、以下のようになります。

[0]	[1]	[2]	[3]	[4]	[5]
237	30	239	40	240	50
モンスター0 キャラクター コード	モンスター0 HP	モンスター1 キャラクター コード	モンスター1 HP	モンスター2 キャラクター コード	モンスター2 HP

★70行の数字を変えると、モンスターのキャラクターやHPが変わります。いろいろ試してみましょう。

80 行で、モンスターのキャラクターコードの位置（0、2、4 のどれか）を乱数で設定しています。

```
80 M=RND(3)*2
```

「**RND(3)**」で 0、1、2 のどれかの数が出るので、「**RND(3)*2**」で 0、2、4 のどれかになります。

90 行で、M を使ってモンスターの HP を取り出しています。

```
90 I=[M+1]+RND(21)
```

M が 0、2、4 の値なので、M+1 は 1、3、5 になり、HP の値が入った配列の番号を指します。

また、配列変数に入った HP の値（30、40、50）をそのまま出しても、毎回同じ HP（＝モンスターの強さ）になって面白くないので、さらに乱数で「**+RND(21)**」として 0～20 のどれかを加算しています。

「SAVE 1」でプログラムを保存しておきましょう。

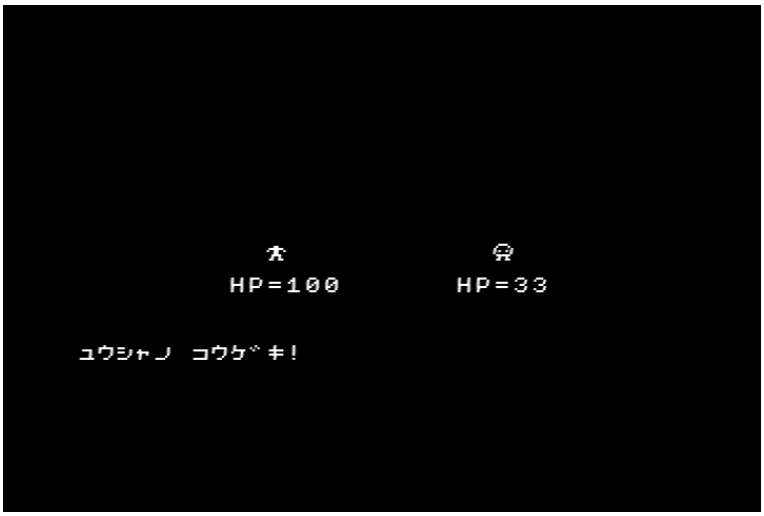
●モンスターとバトルする

バトル画面ができたので、いよいよ勇者とモンスターのバトルを作ります。
まず、勇者→モンスターへの攻撃ターンです。

```
140 LOCATE 0,17
150 PRINT "ﾊﾞﾄﾙ ｽﾀｰﾄ!"
160 WAIT 90
170 '*Y TURN
180 LOCATE 0,17
190 PRINT "ﾕｳｼｬﾉ ｺｳｹﾞｷ!"
200 A=RND(51)
210 I=I-A
220 LOCATE 23,13
230 PRINT I;" "
240 WAIT 90
250 LOCATE 0,17
260 PRINT "                "
```

バトル開始表示
時間待ち
勇者の攻撃表示
攻撃ポイントを乱数で設定
モンスターの HP を攻撃ポイントだけ削る
削られた HP を表示
時間待ち
攻撃表示を消す

プログラムを実行してみましょう。
勇者がモンスターへ攻撃して、
モンスターの HP が減ります。



攻撃の時は、乱数で攻撃ポイントを決めています。

```
200 A=RND(51)
210 I=I-A
```

★200 行の乱数の計算を変えると、勇者の攻撃力が変わります。いろいろ試してみましょう。

続いて、モンスター→勇者への攻撃ターンを作ります。

```
270 '*M TURN
280 LOCATE 0,17
290 PRINT "モンスターノ コウケ゛キ!"
300 A=RND([M+1]/2)
310 H=H-A
320 LOCATE 11,13
330 PRINT H;"  "
340 WAIT 90
350 LOCATE 0,17
360 PRINT "                    "
```

モンスターの攻撃表示

攻撃ポイントを乱数で設定

勇者の HP を攻撃ポイントだけ削る

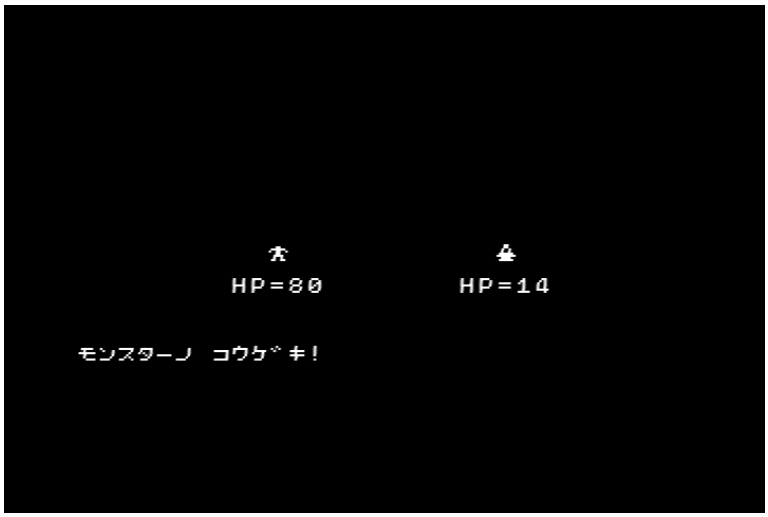
削られた HP を表示

時間待ち

攻撃表示を消す

※勇者の攻撃ターンのプログラムとほとんど一緒なので、そちらのリストを表示してスクリーンエディットすると、入力文字数が少なくて済みます。

プログラムを実行してみましょう。
勇者の攻撃ターンの後、モンスターが勇者に攻撃します。



モンスターの攻撃ポイントは、モンスター自身の HP から決めています。

```
300 A=RND([M+1]/2)
```

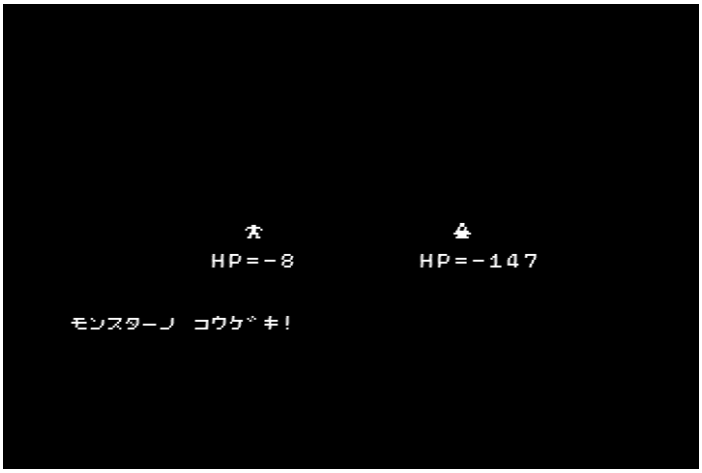
配列変数に設定された HP 初期値から決めているので、初期値が高いほど＝強いモンスターほど、攻撃ポイントも高くなります。

★この計算を変えると、モンスターの攻撃力が変わります。いろいろ試してみましょう。

このままだと、勇者とモンスターが 1 回ずつ攻撃するだけで、バトルが終了してしまいます。
前へ戻って、バトルを続けるようにしましょう。

```
350 LOCATE 0,17
360 PRINT "                    "
370 GOTO 170
```
勇者の攻撃ターンへもどる

プログラムを実行してみましょう。
勇者とモンスターがバトルし続けます。
ただし、お互いの HP がマイナスになっても、バトルを続けてしまいます。



HP がマイナスになってもバトルするのはおかしいので、「HP が 0 以下になった方が負け」と勝ち負け判定を追加します。

```
250 LOCATE 0,17
260 PRINT "                    "
265 IF I<=0 THEN GOTO 380
…(中略)…
350 LOCATE 0,17
360 PRINT "                    "
365 IF H<=0 THEN GOTO 450
370 GOTO 170
380 ´*Y WIN
390 LOCATE 0,17
400 PRINT "ﾕｳｼｬﾉ ｶﾁ!"
410 WAIT 90
420 LOCATE 0,17
430 PRINT "                    "
440 END
…(↓つづく↓)…
```

もしモンスターの HP が 0 以下になったら、勇者が勝った処理へ

もし勇者の HP が 0 以下になったら、モンスターが勝った処理へ

勇者勝利の処理

勇者勝利を表示

時間待ち

勇者勝利を消去

プログラムを終了

```
…(↓つづき↓)…
450 '*M WIN
460 LOCATE 0,17
470 PRINT "モンスターノ カチ!"
480 WAIT 90
490 LOCATE 0,17
500 PRINT "          "
510 END
```

モンスター勝利の処理

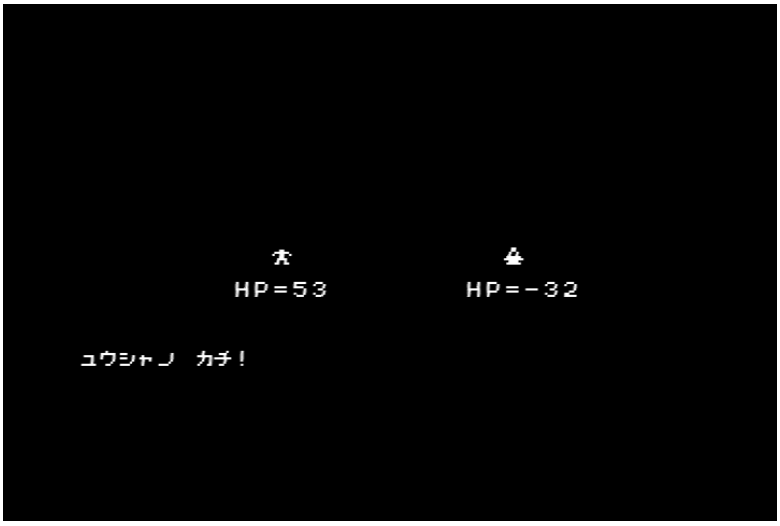
モンスター勝利を表示

時間待ち

モンスター勝利を消去

プログラムを終了

プログラムを実行してみましょう。
勇者とモンスターがバトルして、どちらかのHPが0以下になると、勝負が付きます。
勇者のHPが高く設定されているので、勇者が勝つことが多いでしょう。



勝ち負けの判断は、勇者のHP(変数H)、モンスターのHP(変数I)が0以下になった時に、それぞれが勝った処理へジャンプするようにしています。

```
265 IF I<=0 THEN GOTO 380
…(中略)…
365 IF H<=0 THEN GOTO 450
```

「**I<=0**」は「Iが0より小さいか等しい」、つまり「Iが0以下」という条件式です。
もし「**I<0**」とすると、「Iが0より小さい」となって、Iが0の時は成り立たなくなります。

「SAVE 1」でプログラムを保存しておきましょう。

●効果音をつける

バトルで迫力が出るように、効果音をつけましょう。

```
170 ´*Y TURN
180 LOCATE 0,17
190 PRINT "ユウシャノ コウゲキ!"
195 BEEP 10,5
200 A=RND(51)
210 I=I-A
220 LOCATE 23,13
230 PRINT I;" "
240 WAIT 90
250 LOCATE 0,17
260 PRINT "                "
265 IF I<=0 THEN GOTO 380
270 ´*M TURN
280 LOCATE 0,17
290 PRINT "モンスターノ コウゲキ!"
295 BEEP 30,5
300 A=RND([M+1]/2)
310 H=H-A
320 LOCATE 11,13
330 PRINT H;" "
340 WAIT 90
350 LOCATE 0,17
360 PRINT "                "
365 IF H<=0 THEN GOTO 450
370 GOTO 170
380 ´*Y WIN
390 LOCATE 0,17
400 PRINT "ユウシャノ カチ!"
405 BEEP 10,30
410 WAIT 90
420 LOCATE 0,17
430 PRINT "                "
440 END
…(↓つづく↓)…
```

195 BEEP 10,5 ← 勇者の攻撃の効果音を出す

295 BEEP 30,5 ← モンスターの攻撃の効果音を出す

405 BEEP 10,30 ← 勇者が勝った効果音を出す

```
…(↓つづき↓)…
450 ´*M WIN
460 LOCATE 0,17
470 PRINT "モンスターノ カチ!"
475 BEEP 30,30
480 WAIT 90
490 LOCATE 0,17
500 PRINT "              "
510 END
```

モンスターが勝った効果音を出す

プログラムを実行してみましょう。
バトルの攻撃、終了時に効果音が鳴ります。

音を出すには BEEP（ビープ）命令を使います。BEEP 命令の文法は以下のとおりです。

```
BEEP   30    ,30
       音の高さ  音の長さ
```

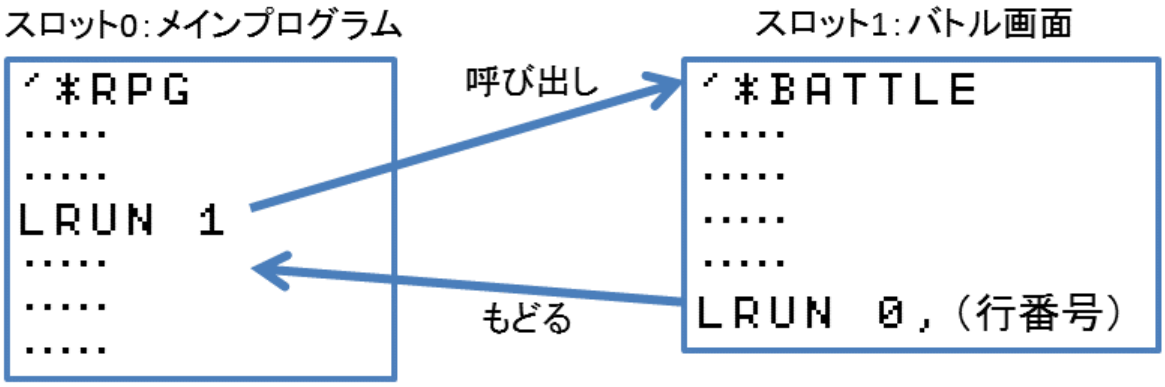
音の高さ	1～255 で指定する。省略可能。
音の長さ	60 分の 1 秒単位で指定する。「60」で 1 秒。省略可能。

★BEEP の数字を変えると、音の高さや長さが変わります。いろいろ変えて試してみましょう。

「SAVE 1」でプログラムを保存しておきましょう。

●メインプログラムからバトル画面を呼び出す

バトル画面のプログラムができたので、メインプログラムから呼び出すようにしましょう。



他のスロットのプログラムへジャンプするには、LRUN(ロードラン)命令を使います。
指定したスロットのプログラムを読み込んで実行します。

```
LRUN  1  ,100
      スロット番号  行番号
```

スロット番号	ジャンプ先のプログラムのスロット番号。
行番号	ジャンプ先のプログラムのどの行から実行するかを指定する。 省略した場合は最初から実行する。

まず、現在編集中のバトル画面プログラムを改造して、バトルが終了したらメインプログラムへもどるようにします。(END でプログラムが終了している行を書きかえます)

```
380 '*Y WIN
390 LOCATE 0,17
400 PRINT "ユウシャノ カチ!"
405 BEEP 10,30
410 WAIT 90
420 LOCATE 0,17
430 PRINT "              "
440 G=1:LRUN 0,190
450 '*M WIN
460 LOCATE 0,17
470 PRINT "モンスターノ カチ!"
475 BEEP 30,30
480 WAIT 90
490 LOCATE 0,17
500 PRINT "              "
510 G=2:LRUN 0,190
```

変数 G を 1 にして、スロット 0 の 190 行へジャンプ

変数 G を 2 にして、スロット 0 の 190 行へジャンプ

変数 G は、メインプログラムにもどった時に、バトルで勇者とモンスターのどちらが勝ったのかを判断するのに使います。

それから、バトルプログラムの先頭で勇者の HP（変数 H）を設定している所を消します。

```
10 '*RPG-BATTLE
20 CLS
30 LOCATE 10,11
```

「：H=100」を消す

これは、メインプログラムの H の値をそのまま引き継がないといけないからです。

メインプログラムの編集へ移ります。
以下の作業を順番どおりにやります。

まちがえると、これまで作ったプログラムが消えてしまう恐れがあります。

注意して作業してください。

（1）今作っているバトル画面プログラムを、スロット 1 に保存します。

```
SAVE 1
Saved 000byte
OK
```

（2）スロット 0 のメインプログラムを呼び出します。

```
LOAD 0
Loaded 000byte
OK
```

※（1）の保存を忘れると、ここでこれまで作ったプログラムが消えてしまいます。

メインプログラムで、モンスターに出会ったらバトル画面へジャンプするようにします。
バトル画面から戻って、もしモンスターが勝ったら、ゲームオーバーになるようにします。

```
100 '*GAMELOOP
…(中略)…
160 C=SCR(U,V)
170 IF C=236 THEN GOSUB 400
180 G=0:IF RND(100)=0 THEN LRUN 1
190 IF G>0 THEN GOSUB 300
195 IF G=2 THEN GOTO 270
200 LOCATE X,Y
210 PRINT " ";
220 LOCATE U,V
230 PRINT CHR$(250);
240 X=U:Y=V
250 WAIT 5
260 GOTO 100
270 '*GAMEOVER
280 LOCATE 10,12
285 PRINT "*GAME OVER*"
290 END
```

100 分の 1 の確率でモンスターとバトル

バトルが終わると、この行へもどるマップを再度表示

モンスターが勝ったらゲームオーバーへジャンプ

ゲームオーバーを表示

実行する前に、必ず「SAVE 0」でプログラムを保存してください。

```
SAVE 0
Saved OOObyte
OK
```

※保存しないで実行すると、このページで入力したプログラムが消えてしまいます。

「RUN」でプログラムを実行してください。
マップ画面でしばらく時間が経つと、モンスターに出会ってバトルになります。
バトルが終わると、勇者が勝った場合はそのままゲームが続きます。
モンスターが勝った場合はゲームオーバーになります。

※プログラムを ESC キーで止める時は、必ずマップ画面で止めてください。
　バトル画面で止めると、バトルプログラムが呼び出されたままになります。

●おにぎりを表示する

このままだと、バトルで勇者の HP が減る一方なので、HP を回復させるアイテム「おにぎり」を登場させましょう。基本的には、村人と同様のプログラムです。
まず、プログラムの最初で、おにぎりの位置を設定します。

```
10 '*RPG
20 CLV
30 H=100
40 X=15:Y=11
42 P=RND(32):Q=RND(23)
44 D=RND(32):E=RND(23)
50 GOSUB 300
```

おにぎりの座標(D,E)を乱数で設定

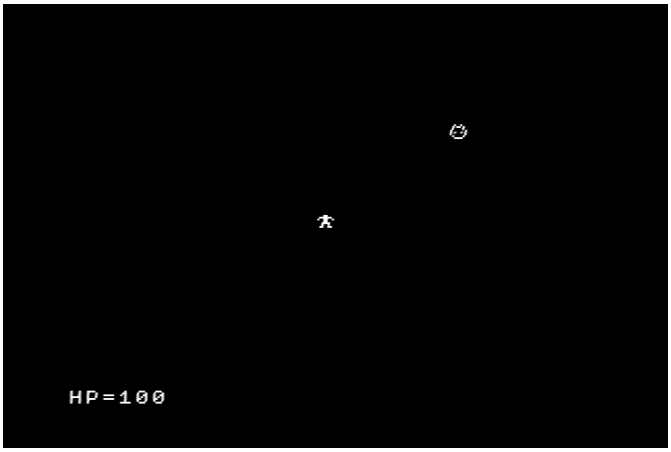
マップを表示するサブルーチンを改造して、おにぎりを表示するようにします。

```
300 '*MAP
310 CLS
320 LOCATE 0,23
330 PRINT "HP=";H;
340 LOCATE X,Y
350 PRINT CHR$(250);
360 LOCATE P,Q
365 PRINT CHR$(236);
370 LOCATE D,E
375 PRINT CHR$(235);
390 RETURN
```

座標(D,E)におにぎりを表示

実行する前に、必ず「SAVE 0」でプログラムを保存してください。

プログラムを実行してみましょう。
おにぎりが画面に表示されます。
プログラムを実行するたびに、おにぎりの表示位置が変わります。



●おにぎりで HP を回復させる

勇者がおにぎりをゲットしたら、HP が回復するようにしましょう。
ゲームループの中で、勇者がおにぎりと出会ったかどうかを判断します。

```
100 '*GAMELOOP
…(中略)…
160 C=SCR(U,V)
170 IF C=236 THEN GOSUB 400
175 IF C=235 THEN GOSUB 500
180 IF RND(100)=0 THEN LRUN 1
```

勇者の行き先がおにぎりなら
おにぎりのサブルーチンを呼ぶ

500 行から、おにぎりの処理をするサブルーチンを追加します。

```
500 '*ONIGIRI
510 LOCATE 12,23
520 PRINT "ｵﾆｷﾞﾘ ｹﾞｯﾄ!"
530 WAIT 120
540 H=100
550 D=RND(32):E=RND(23)
560 GOSUB 300
570 RETURN
```

コメント
画面下に「オニギリ　ゲット！」と表示
時間待ち
勇者の HP を 100 に回復
おにぎりの座標(D,E)を再設定
マップ表示サブルーチンを呼ぶ
→HP とおにぎりが再表示される
もどる

実行する前に、必ず「SAVE 0」でプログラムを保存してください。

プログラムを実行してみましょう。
勇者がバトルして HP が減った後、おにぎりをゲットすると、
HP が 100 に回復します。