

ニンテンドー3DSで ゲームプログラムを作ろう！

上田市マルチメディア情報センター
春休みプログラムセミナー2017

 SmileBoom Co.Ltd.

■本日の予定

- 10:00 スタッフ紹介、スケジュール説明
- 10:10 プチコンプログラミング
- 12:00 昼休み
- 13:00 プチコンプログラミング続き
- 15:50 休憩
- 16:00 ゲームクリエイターになるためには
- 17:00 参加者アンケート記入、事務連絡、解散

<講師>

株式会社スマイルブーム 小林貴樹

ニンテンドー3DSでゲーム開発

プチコンの使い方



■プチコン3号入手方法

- プチコン3号はニンテンドーeショップでダウンロード販売されています
- ショップ画面から「プチコン3号」で検索すると見つけることができます
 - 消費税込みで1000円
- スマイルブームの公式ページには初級講座などの情報があります
 - <http://smilebasic.com/>

SMILE BASIC  JP ▼

最新ニュース

2017.3.16
【ニンテンドーDSi LL/DSiをお持ちで「プチコンmkII」の購入を検討されている方へ】
2017.3.31でニンテンドーDSiショップのサービスが終了となります。ニンテンドーDSi LL/DSiをお持ちのお客様で、プチコンmkIIの購入をご検討されている方は、2017.3.31以降の購入はできなくなりますので早めにお買い求めください。詳しくは [こちらのページ](#)をご確認ください。

2017.3.16
『プチコン』シリーズ 公式マガジン第2弾『SMILEBASIC MAGAZINE SPECIAL』が3月16日に発売となりました！くわしくは『SMILEBASIC MAGAZINE』詳細ページをごらんください。
ニンテンドー3DS向けの『プチコン3号』および Wii U 向けの『プチコン BIG』公式攻略ガイドとして、すべての命令リファレンスを収録！
さらに、Windows 向け 3D RPG作成ソフト『SMILE GAME BUILDER』の利用ガイドも併録しています。
お求めは、Amazon.co.jp および全国の各書店にて。

@PetitComputer
プチコン
@PetitComputer
【お知らせ】
2017.3.31でニンテンドーDSiショップのサービスが終了となります。プチコンmkIIの購入を検討されている方は早めにお買い求めください。(3DSからは引き続き購入可能です)。詳細は[こちらのページ](http://smilebasic.com/library/picm2/)をご確認ください。
smilebasic.com/library/picm2/
3月16日

SmileBASIC 講座
簡易電子説明書
命令表
障害情報
更新データ
サポート

■ プチコンの画面と操作方法

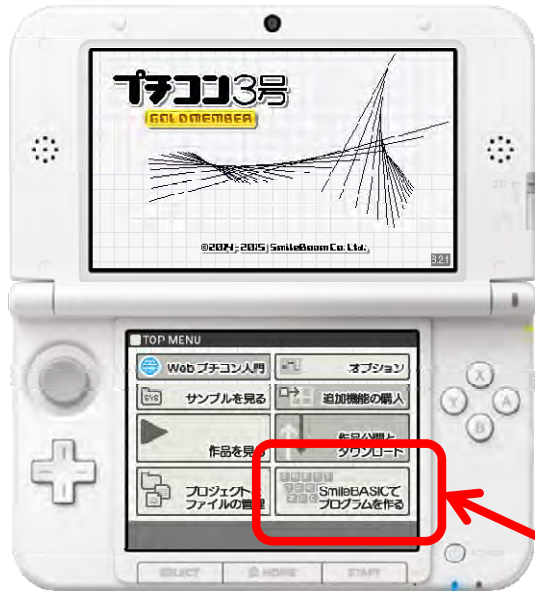
トップメニュー

- **TOPMENU** の「SmileBASICでプログラムを作る」ボタンからBASICを起動
- プログラムを実行するときは、^{タイル}**DIRECT** ボタンでコンソール表示に切り替えます
- プログラムを書くときは、^{エディット}**EDIT** ボタンで編集モードに切り替えます

SmileBASICでプログラム開発中の切り替え

DIRECT モードでプログラムを実行

EDIT モードでテキストを編集



TOPMENU との行き来



DIRECT と EDIT の切り替え



ツールへ

■プチコンのキーボードについて

- SmileBASICの作業中は下画面にキーボードが表示されます



命令候補エリア

- 通常は命令候補を表示
- #で始まる定数の確認

EDIT中：ページ単位移動

EDIT中：カーソル移動

●CAP

- SHIFTキーを押したままの状態にする

文字種類の変更

- 英数字
- 記号（欧州文字）
- ひらがな/カタカナ

ファンクションキー

- KEY命令で再定義可能なボタン



スマイルボタン

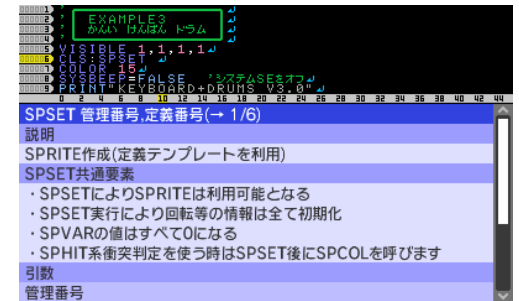
- ツールを実行する

STARTボタン

- プログラム実行と停止



スペースキー（1文字分の空白）



検索モード切替

ヘルプ表示モード切替

Yボタン

- 1文字削除(BSキー相当)

Aボタン

- 改行(エンターキー相当)

INS●


- 挿入と上書きの切り替え

編集機能

- 範囲選択付きのコピーやペーストなど



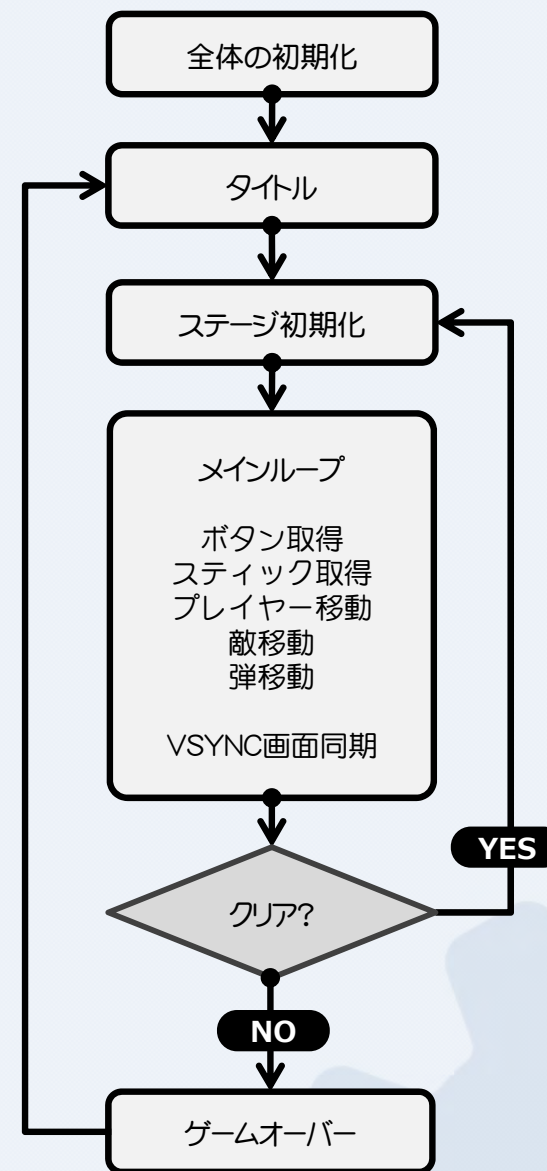
エンターキー（改行）

本資料内で、 という記号がある場合は、エンターキーを入力すると解釈してください。



砲台からの弾で敵を倒すゲーム

タンクバトルゲーム！



■テンプレート（教材）を公開キー経由で持ってくる

- 「作品公開とダウンロード」を押し「公開キーを使ってダウンロード」を押す



公開キー
12WEDNAE

- 作業用のプロジェクトフォルダを「UEDA2017」に変更する



- 「SmileBASICでプログラムを作る」を押してBASICに入る



■BASICでテンプレート（教材）を読み込む

- Lボタンを押しながらLOADを押して一覧から「TANK1」を選んで「決定」



-  エディットボタンを押してプログラムが読み込まれていたら成功です。

```
00001 うえだ2017 @SmileBoom
00002
00003
00004 OPTION STRICT
00005 VAR I, J, M, X, Y, IX, VX, VY
00006 VAR BN, BI, BR, ホンゾウ
00007 VAR STX, STY, ステッカー
00008 VAR DB=0, ティック
00009 VAR FM=0, フィン
00010 VAR TM=0, テン
00011 VAR ST=0, ステ
00012 VAR MS=0, モス
00013 VAR HI=58, SC=0, ハイスコア, スコア
00014 VAR _VX=0, _VY=1, _HP=2
00015 VAR _BC=7
00016
00017
00018 PLCNT=
00019 PLTOP=
00020 PLEND=PLTOP+PLCNT-1
00021 PLDEF=2302
00022
00023 PSCNT=
00024 PSTOP=PLEND+1
00025 PSEND=PSTOP+PSCNT-1
00026
00027 ITCNT=
00028 ITTOP=PSEND+1
00029 ITEND=ITTOP+ITCNT-1
```

★ヒント

コンソール画面から直接LOAD命令で読み込む方法もあります。

```
SmileBASIC for Nintendo 3DS ver 3.5.2
(C)2011-2017 SmileBoom Co. Ltd.
8318976 bytes free

[UEDA2017]OK
LOAD "TANK1"
```

■この資料の説明と検索、途中経過のSAVE

- 1ページごとに以下のようにプログラムと説明が書かれています

1	1	
2	2	うめぞ2017@SmileBoom
3	3	
4	4	OPTION STRICT

← 薄いピンク色になっている部分は、最初から用意されているので「入力不要」です。

番号 **プログラム** **プログラムや変数などの説明**

※ 番号部分は、入力した時の改行の違い等でずれることがありますが目安なので気にしないでください

- プログラムが長くなって探しにくいときは虫眼鏡アイコンを使います

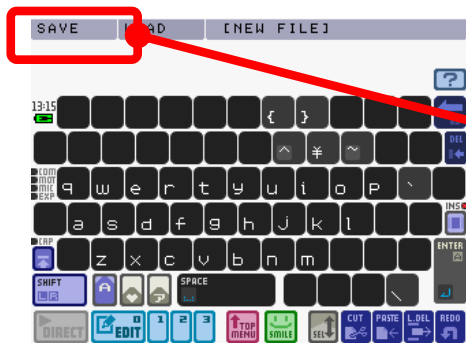
- ページごとに書かれている文字列を入れてジャンプしてください



- 電池切れでせっかく入力したプログラムが消えないように時々SAVEします

Lボタンを押しながらSAVEを押す

名前を付けて保存



★ヒント

SAVEするときの名前は、毎回違う名前にしておくと便利です。1ページの入力が終わるごとに保存。

TANK1245 のように、TANKと保存する時の時間で名前を付けます

それでは、さっそく次のページからプログラムの入力を始めましょう！

■ゲーム全体で利用する変数の定義

```
1 *  
2 * うえだ2017@SmileBoom  
3 *  
4 OPTION STRICT  
5 VAR C, I, J, M, X, Y, IX, VX, VY  
6 VAR BN, BI, BR 'ボタンよう  
7 VAR STX, STY 'スティックよう  
8 VAR DB=0 'デバックよう  
9 VAR EW=0 'じしんよう  
10 VAR TM=0 'じかん  
11 VAR ST=0 'ステージ  
12 VAR MS=0 'もしゃうじ  
13 VAR HI=58, SC=0 'ハイスコア、スコア  
14 ' --- スプライトへんすう  
15 VAR _VX=0, _VY=1, _HP=2  
16 VAR _BC=7
```

ゲーム全体で利用する変数や定数を定義。

BASICでは、文字入力数を減らすために、変数名を短くして使うことがあります。

BN → BUTTON NEWを短縮

BI → BUTTON IMPACTを短縮

※長く書いても問題なく動きます

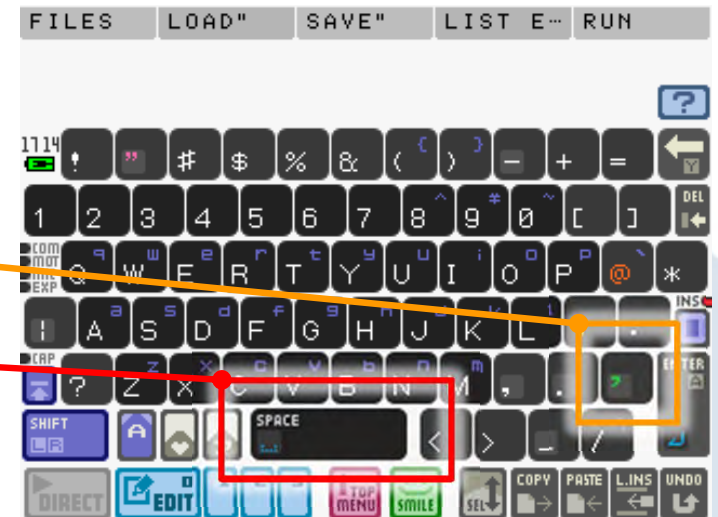
※途中にスペースは入れられません

★ヒント

コメント入力用の記号「*」は、
シングルクォーテーションと呼びます。
コメント記号以降の文字は緑になります。

```
VAR DB=0 'デバックよう  
VAR EW=0 'じしんよう
```

グレーの四角「□」はスペース記号（空白文字）です。
この文字は、キーボードの長いボタンを押して入力します。



■スプライトを管理する番号や数などの定義

```
17 * ---□プレイヤー-よう
18 VAR□PLCNT=□2
19 VAR□PLTOP=□0
20 VAR□PLEND=PLTOP+PLCNT-1
21 VAR□PLDEF=2302
22 * ---□プレイヤーのタマよう
23 VAR□PSCNT=□3
24 VAR□PSTOP=PLEND+1
25 VAR□PSEND=PSTOP+PSCNT-1
26 * ---□アイテムよう
27 VAR□ITCNT=□10
28 VAR□ITTOP=PSEND+1
29 VAR□ITEND=ITTOP+ITCNT-1
30 * ---□もじよう
31 VAR□TXCNT=□32
32 VAR□TXTOP=ITEND+1
33 VAR□TXEND=TXTOP+TXCNT-1
34 * ---□テキよう
35 VAR□ENCNT=□30
36 VAR□ENTOP=TXEND+1
37 VAR□ENEND=ENTOP+ENCNT-1
38 VAR□ENDEF=3024
39 VAR□EC=0
40 * ---□てきのたまよう
41 VAR□ESCNT=□20
42 VAR□ESTOP=ENEND+1
43 VAR□ESEND=ESTOP+ESCNT-1
44 * ---□ぼくはつよう
45 VAR□BMCNT=□30
46 VAR□BMTOP=ESEND+1
47 VAR□BMEND=BMTOP+BMCNT-1
```

スプライトを使って表示物を管理するための定数の集まりです。こちらにも短縮名称を使っています。

PLCNT → PLAYERCOUNT
PLTOP → PLAYERTOP
PLEND → PLAYEREND
PLDEF → PLAYER SPDEF

PS で始まる定数はプレイヤーの弾用、
IT で始まる定数はアイテム用、
TX で始まる定数は文字列表示用、
EN で始まる定数は敵用、
ES で始まる定数は敵の弾用、
BM で始まる定数は爆発用、

<例>

一度に出現できる敵の弾の数を増やしたいときは、ENCNTの数を30以上にします。

プレイヤーの弾の数を増やしたいときは、PSCNTの数を3以上にします。

プログラムの様々な場所で使われる可能性がある数値については、直接数字で書かずに一度定数として定義しておいてから利用すると、後で数を増やしたり減らしたりする際に楽です。

■画面の初期化、タイトル表示

GAMELOOP



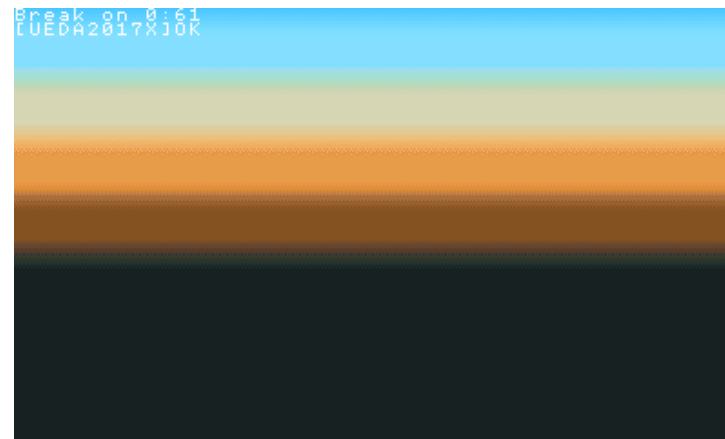
```
48 '----ひょうじのしよきか↓
49 XSCREEN 0, 256, 2:CLS↓
50 GCLS:GCLIP 1, 0, 0, 399, 239:GPRI0 100↓
51 BGSCREEN 0, 32, 32:BG0FS 0, 0, 0, 1024↓
52 '----フォントへんこう↓
53 'LOAD "GRPF:FONT", FALSE↓
54 '---- BGではいけいをつくる↓
55 FOR X=0 TO 31↓
56   FOR Y=0 TO 15↓
57     IF Y<10 THEN C=&H25F+Y*32 ELSE C=&H37F↓
58     BGPOT 0, X, Y, C↓
59   NEXT Y↓
60 NEXT X↓
61 ↓
62 '====↓
63 'タイトル↓
64 '====↓
65 @GAMELOOP↓
66 EFCSET 0:DISPCLR↓
67 TITLE↓
68 ST=0:SC=0↓
69 ↓
```

XSCREEN で上画面だけを使うことを宣言。

※フォントを外部で作ったときの読み込み例

BG用に32x32のスクリーンサイズを定義して空の絵を表示。

画面を消してタイトル画面表示を呼び出す。



ここまでの入力ではタイトル画面は表示されません

動くかどうか試したいときは、STARTボタンを押してみましょう

■ステージの準備、メインループ

STAGELoop



```
70 ?====  
71 ?ゲーム  
72 ?====  
73 @STAGELoop  
74 EFCSET 2:DISPCLR  
75 SPCLR:GCLS:CLS  
76 MAKESTAGE ST  
77 COLOR#TWHITE, #TBLACK  
78 PRINT" " *100  
79 COLOR#TWHITE, 0  
80 PUTSCORE  
81 SETTEXT 121, "STAGE" +STR$(ST+1)  
82  
83 ?=====  
84 ?メインループ  
85 ?=====  
86 VAR GM=1  
87 WHILE GM>0 && EC!=0  
88 BN=BUTTON(0)  
89 BI=BUTTON(2)  
90 BR=BUTTON(3)  
91 STICK OUT STX, STY  
92 CALL SFRITE  
93 PUTSCORE  
94 EARTHQUAKE  
95 VSYNC 1  
96 WEND  
97
```

ステージごとの初期化を行うプログラム。

画面を消して、ステージごとのマップを表示。
ゲーム中のスコア表示用の背景を表示。
ステージ名をスプライトとして表示。

ゲームのメインループ（ステージクリアするかゲームオーバーになるまで繰り返しま）。

ボタン押下情報取得、スティックの状態取得、
スプライトの処理を呼び出し、スコアも表示。
画面の揺れプログラムの呼び出し

ここまでの入力ではゲーム画面などの表示は増えません

■ ステージクリア、ゲームオーバー

GAMEOVER



```
98 ?=====↓
99 ?クリア/おわり↓
100 ?=====↓
101 BGMSTOP:EFCSET 0↓
102 IF GM==0 THEN @GAMEOVER↓
103 ↓
104 ?---- CLEAR↓
105 SETTEXT 113, "STAGE CLEAR" ↓
106 WHILE MS>0 ↓
107   CALL SPRITE ↓
108   VSYNC 1 ↓
109 WEND ↓
110 ST=ST+1 ↓
111 GOTO @STAGELoop ↓
112 ↓
113 ?--- GAMEOVER ↓
114 @GAMEOVER ↓
115 SETTEXT 122, "GAME OVER" ↓
116 WHILE MS>0 ↓
117   FOR I=0 TO PLCNT-1 ↓
118     IF SPUSED(I+PLTOP)==TRUE THEN ↓
119       SPOFS I+PLTOP OUT X, Y ↓
120       C=RGB(RND(100)+64, 0, 0) ↓
121       GLINE X, Y, X+RND(40)-20, Y-RND(80), C ↓
122     ENDIF ↓
123   NEXT ↓
124   CALL SPRITE ↓
125   VSYNC 1 ↓
126 WEND ↓
127 GOTO @GAMELOOP ↓
128 ↓
```

GMの値が0だったらゲームオーバー。

ステージクリアの時は、文字表示が終わるまで待つて次のステージへ (@STAGELoopへ飛ぶ)

ゲームオーバーの時は、プレイヤーに赤い線の爆発演出を表示してタイトル画面へ (@GAMELOOPへ飛ぶ)

M1 ~ **M2** のページを入力すると動きのある文字が表示されます

ここまでの入力では特に新しい表示の追加はありません

動くかどうか試したいときは、STARTボタンを押してみましょう

■画面を消す、スコアボードの表示

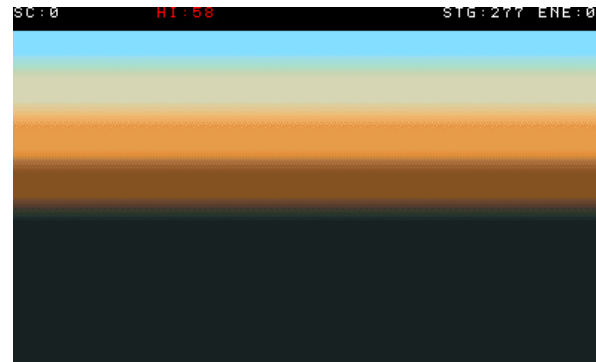
PUTSCORE



```
129 * がめんしょうきょ ↵
130 * がめんしょうきょ ↵
131 * がめんしょうきょ ↵
132 DEF □DISPCLR ↵
133 SPCLR ↵
134 GCLS □RGB(0, 0, 0, 0) ↵
135 COLOR □#TWHITE, 0 ↵
136 CLS ↵
137 END ↵
138 ↵
139 * スコアボード ↵
140 * スコアボード ↵
141 * スコアボード ↵
142 DEF □PUTSCORE ↵
143 COLOR □#TWHITE, #TBLACK ↵
144 LOCATE □0, 0:PRINT □"SC:";SC ↵
145 LOCATE □36, 0:PRINT □"STG:";ST+1 ↵
146 LOCATE □44, 0:PRINT □"ENE:";EC ↵
147 COLOR □#TRED ↵
148 LOCATE □12, 0:PRINT □"HI:";HI ↵
149 COLOR □#TWHITE, 0 ↵
150 END ↵
151 ↵
```

利用している画面の情報をクリアします。
文字の色を白に戻します。

スコア、ハイスコア、敵の数、ステージ数の表示



画面の上の方に黒い帯とスコアなどの数字が表示されます

動くかどうか試したいときは、STARTボタンを押してみましょう

T1

■タイトル画面

TITLE



```

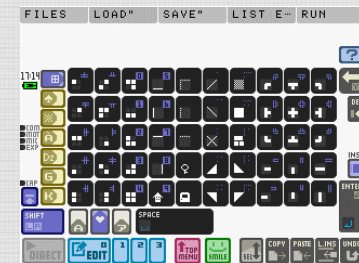
152 * [タイトル]
153 *
154 *
155 DEF TITLE
156 VAR X=15, Y=10
157 COLOR #TYELLOW, 0
158 LOCATE X, Y+0: PRINT "          "
159 LOCATE X, Y+1: PRINT "          "
160 LOCATE X, Y+2: PRINT " [          ] "
161 LOCATE X, Y+3: PRINT " [          ] "
162 LOCATE X, Y+4: PRINT " [          ] "
163 LOCATE X, Y+5: PRINT " [          ] "
164 LOCATE X, Y+6: PRINT " うえだしまるメディアじょうほうセンター "
165 COLOR #TWHITE, 0
166 * --- びんりの丸をかく
167 GPRIO -10
168 FOR I=0 TO 8
169   X=96+I*12
170   GCIRCLE 200, 120, X, RGB(0, 255-I*16, 0)
171   GPAINT 200+X+1, 120, RGB(0, 255-I*16, 0)
172   GPAINT 200-X-1, 120, RGB(0, 255-I*16, 0)
173 NEXT I
174 * --- タマをだしながらボタンまち
175 BGMPLAY 40
176 LOCATE 20, 22
177 PRINT " ボタンをおしてください "
178 ANYKEY "TITSUB"
179 BGMSTOP
180 SPCLR:CLS:GCLS:GPRIO 100
181 END
182
183 * [タイトルでたまはっしゅ]
184 *
185 *
186 DEF TITSUB
187 VAR X, Y, R=MAINCNT, P
188 IF RND(5)>0 THEN RETURN
189 R=RND(180)-90
190 P=RND(10)+4
191 SETSHOT PSTOP, PSEND, 2292, 200, 80, R, P
192 END
193

```

記号で作られたタイトル画面の表示
グラフィック画面に丸を書く
弾を発射しながら何かボタンの入力を待つ



タンクバトルの文字を書く記号は
ハートマークを押して切り替えます。



画面に緑色の丸い縁を描く

タイトル画面で爆弾を飛ばす

タイトルロゴの上の発射口から爆弾を発射

ここまでのプログラムでは、
まだ画面に爆弾は表示されません。



■共通：ボタンが押されるまで待つ

ANYKEY



```
194 '
195 ' なにがボタンが押されるまで待つ
196 '
197 ' FUNC#:ユーザージョリ
198 '
199 DEF ANYKEY FUNC#
200 VSYNC 1
201 WHILE 1
202 IF BUTTON(2) THEN BREAK
203 IF FUNC# != "" THEN CALL FUNC#
204 CALL SPRITE
205 VSYNC 1
206 WEND
207 END
208
```

何かボタンが押されるまで待つ共通処理。
待っている間、他のプログラムを実行できる。

■グラフィック画面揺らし、緑色の調査

GETCOL



```
209 * じめんをゆらす ↵
210 * じめんをゆらす ↵
211 * じめんをゆらす ↵
212 DEF EARTHQUAKE ↵
213 IF EW==0 THEN RETURN ↵
214 EW=EW-1 ↵
215 IF EW>0 THEN GOF S0, RND(8)-4 ELSE GOF S0, 0 ↵
216 END ↵
217 ↵
218 * カベのいろはんでい(GREEN) ↵
219 * カベのいろはんでい(GREEN) ↵
220 * カベのいろはんでい(GREEN) ↵
221 * X, Y: けんさぎひょう ↵
222 * ----- ↵
223 * 0いけいのときかべあり ↵
224 * ↵
225 DEF GETCOL(X, Y) ↵
226 VAR OX, OY, C=0 ↵
227 FOR OY=0 TO 1 ↵
228   FOR OX=0 TO 1 ↵
229     C=C OR GSPoit(X+OX, Y+OY-1) ↵
230   NEXT ↵
231 NEXT ↵
232 RETURN C AND &H00FF00 ↵
233 END ↵
234 ↵
```

一定時間グラフィック画面を上下に揺らす

グラフィック画面の色を調べる。
緑要素が入っている場合壁扱いになる。

GSPoitからの戻り値は、ARGBの要素が各8ビットずつの合成された32ビット情報となる。緑の要素は8ビット目から15ビット目に格納されている。

B1 ~ **B4** のページを入力するとタイトルで爆弾が飛び回ります

ここまでの入力では特に新しい表示の追加はありません

動くかどうか試したいときは、STARTボタンを押してみましょう

T3

■プレイヤー移動プログラム

PLAYER



```
235 *
236 * プレイヤー
237 *
238 DEF PLAYER
239 VAR R, X, Y, VX=0, VY
240 VAR IX=CALLIDX
241 * --- 作られはんでいく(デキとデキのタマ)
242 VAR DD=SPHITSP(IX, ENTOP, ESEND)
243 IF GM==0 THEN RETURN
244 IF DD>-1 THEN
245   SPVAR DD, _HP, 0 'あいてをけす
246   GM=0: BEEP 114
247   RETURN
248 ENDIF
249 * --- ぶやうにすすむ(スティック)
250 IF STX<-0.2 THEN VX=-1
251 IF STX> 0.2 THEN VX=VX+1
252 SPVAR IX, _VX, VX
253 IF MOV OBJ(IX, 1, 0) == -1 THEN GM=0 'しぼう
254 * --- かくとへんか(砲)
255 R=SPROT(IX)
256 VX=0
257 IF BN AND (#RIGHT OR #R) THEN VX=VX+1
258 IF BN AND (#LEFT OR #L) THEN VX=VX-1
259 * --- かくとへんかちゆう?
260 IF (SPCHK(IX) AND #CHKR) == 0 THEN
261   R=R+VX*3
262   IF R<-60 THEN R=-60
263   IF R> 60 THEN R= 60
264 * --- あたらしい かくと
265 SPANIM IX, "R", -4, R, 1
266 ENDIF
267 * --- タマはっしゅ?
268 VAR BC=SPVAR(IX, _BC)
269 GETOBJ IX OUT X, Y, VX, VY
270 IF VY==0 && (BR AND #A) THEN
271   SETSHOT PSTOP, PSEND, 292, X, Y, R, BC
272 ENDIF
273 IF BN AND #A THEN BC=BC+1 ELSE BC=0
274 SPVAR IX, _BC, BC
275 END
276
```

遊んでみて操作しにくいと感じたら、スティックと十字キーを入れ替えても良いです。

敵や敵の弾とのあたり判定。

ENTOP は、敵の sprite 番号の始まり、ESEND は、敵の弾の_sprite 番号の終わり。これらの番号は連続しているので同時に調査。

スティックによる左右の移動。

十字キーの左右を使った砲台の角度調整。

角度変化していなければ新しい角度を登録。

Aボタンが押されたら弾発射。

X1 と S1 ~ S5 を入力しないと

プレイヤーは表示されません。

P1

■ゲーム中に弾を新規作成

SETSHOT



```
277 *  
278 * タマはっしゅ  
279 *  
280 * □T:さいしょのぼんごう  
281 * □C:さいだいですう  
282 * □N:SPDEFはんだう  
283 * □X, Y:はっしゅびょう  
284 * □R:かくど  
285 * □PW:うちだしつよさ  
286 *  
287 DEF□SETSHOT□T, C, N, X, Y, R, PW  
288 * --- もしひょうじちゅうは、うてない  
289 IF□MS>0□THEN□RETURN  
290 * --- あきかくほ  
291 VAR□IX=NEWOBJ(□T, C, N, "SHOT"□)  
292 IF□IX== -1□THEN□RETURN  
293 * --- しょぞくど  
294 VAR□V=2+(□PW*2)/4  
295 * IF□V>8□THEN□V=8  
296 VAR□VX=SIN(RAD(□R))* V□'しょぞくX  
297 VAR□VY=COS(RAD(□R))* -V□'しょぞくY  
298 SETOBJ□IX, □X, Y, □VX, VY  
299 * --- アニメとうろく  
300 BEEP□120  
301 SPANIM□IX, "I+", -8, 1, 0  
302 SPANIM□IX, "S", -8, 1.5, 1.5, 0  
303 SPROT□IX, R  
304 END  
305 *
```




指定された番号から、指定された個数分までの間で弾を生成する。空きがない場合は弾は出てこない。

B1

■弾移動プログラム

SHOT



```
306 *  ↵
307 *  ↵
308 *  ↵
309 DEF SHOT ↵
310 VAR IX=CALLIDX ↵
311 * --- いどう ↵
312 VAR X, Y, VX, VY ↵
313 GETOBJ IX OUT X, Y, VX, VY ↵
314 VAR NX=X+VX ↵
315 VAR NY=Y+VY ↵
316 VY=VY+0.4: IF VY>8 THEN VY=8 ↵
317 SETOBJ IX, NX, NY, VX, VY ↵
318 * --- 消えはんでい ↵
319 IF SPVAR(IX, _HP)==0 THEN ↵
320   SPCLR IX ↵
321   RETURN ↵
322 ENDIF ↵
323 * --- がめんがい? ↵
324 IF X>-16 && X<400+16 THEN ↵
325   IF Y<240+16 THEN ↵
326     * --- かべ?(ラインはんでい) ↵
327     IF GETCOL(X, Y)==0 THEN ↵
328       IF IX<ENTOP THEN C=#RED ELSE C=#BLUE ↵
329       IF DB THEN GFILL X, Y, X+1, Y+1, C ↵
330       RETURN ↵
331     ENDIF ↵
332     * --- まなま ↵
333     GBOMB X, Y, RND(12)+4, RGB(0, 0, 0) ↵
334   ENDIF ↵
335 ENDIF ↵
336 * --- しぼ ↵
337 SPCLR IX ↵
338 END ↵
339 ↵
```

撃ちだされた弾の移動。
敵も味方も共通の処理として移動する。

体力が0になったら消える。

画面外に出たら消える。

DBが0以外だったらデバッグ用に点を打つ。

着地して爆発する。

B2

■爆発プログラム、爆発用塗りつぶし円表示

GBOMB



```
340 *
341 *   ばくはつ
342 *
343 *   □CX, CY: ぎひょう
344 *   □R: はんけい
345 *
346 DEF□GBOMB□CX, CY, R, C
347 VAR□IX, X1, Y1, X2, Y2, X3, Y3
348 * --- □アニメ
349 BEEP□13, -RND(10)*200
350 SPSET□BMTOP, BMEND, 3428□OUT□IX
351 SPDFS□IX, CX, CY
352 SPANIM□IX, "I+", -16, 3, 1
353 SPFUNC□IX, "FIRE"
354 * --- □おなほり
355 GCIRCLEF□CX, CY, R, C
356 EW=8
357 END
358
359 *
360 *   ぬりつぶしたえん
361 *
362 *   □X, Y: ちゅうしんぎひょう
363 *   □R: はんけい
364 *   □C: いろ
365 *
366 DEF□GCIRCLEF□X, Y, R, C
367 VAR□IX, X1, Y1, X2, Y2, X3, Y3
368 FOR□IX=0□TO□360□STEP□30
369 □X1=X+SIN(RAD(IX))*R
370 □Y1=Y+COS(RAD(IX))*R
371 □X2=X+SIN(RAD(IX)+60)*R
372 □Y2=Y+COS(RAD(IX)+60)*R
373 □X3=X+SIN(RAD(IX)+120)*R
374 □Y3=Y+COS(RAD(IX)+120)*R
375 □GTRI□X1, Y1, X2, Y2, X3, Y3, C
376 NEXT
377 END
378
```

爆発アニメーションの表示。
グラフィック画面に爆発による抜き丸を描く。

グラフィック画面を丸く塗りつぶす。

STEP の後ろの数値を小さくするときれいな丸に近づく。

B3

■炎アニメ終了確認

FIRE



```
379 '
380 ' ほのお
381 '
382 DEF FIRE
383 VAR IX=CALLIDX
384 IF SPCHK(IX) AND #CHKI THEN RETURN
385 SPCLR(IX)
386 END
387
```

爆発後に出る炎のアニメーションが終わったかどうかを判断する。



B1 ~ **B4** のページを入力するとタイトルで爆弾が飛び回ります

動くかどうか試したいときは、STARTボタンを押してみましょう

B4

■敵移動プログラム

ENEMY



```
388 *  
389 *   [テキ]   
390 *  
391 DEF ENEMY   
392 VAR I, R, X, Y, VX, VY, OVX   
393 VAR IX=CALLIDX   
394 * --- VXほかん   
395 OVX=SPVAR(IX, _VX)   
396 * --- かわらばんでいく(プレイヤーのママ)   
397 VAR DD=SPHITSP(IX, PSTOP, PSEND)   
398 IF DD>-1 THEN   
399   SPVAR DD, _HP, 0 'あひてあけず   
400   BEEP 14   
401   GETOBJ IX OUT X, Y, VX, VY   
402   GBOMB X, Y, 24, #LIME 'おどろはくはつ   
403   ENDIF   
404 * --- いどう   
405 IF DD!=-1 || MOVOBJ(IX, 1, 1)==-1 THEN   
406   ' --- しぼう   
407   EC=EC-1   
408   SPCLR IX   
409   RETURN   
410   ENDIF   
411 * --- さひしんのじょうほうしゅどく   
412 GETOBJ IX OUT X, Y, VX, VY   
413 IF OVX!=VX THEN   
414   ANIMOBJ IX, 3024, VX   
415   ENDIF   
416 * --- たまはらっしゅ?   
417 IF VY==0 && (RND(30)==1) THEN   
418   R=RND(90)-45: I=RND(12)+2   
419   SETSHOT ESTOP, ESEND, 3394, X, Y, R, I   
420   ENDIF   
421 END   
422
```

敵の移動。

プレイヤーの弾に当たったら死亡。
死ぬ間にグラフィック画面に緑色の丸を描く。

死亡したのでスプライトを消す。

適当な周期で適当な強さとスピードの弾を発射。
RND(30)の30部分を少なくすると弾が多く出る。

★敵の移動処理

E1

■ アイテム移動、スコア加算

ITEM



```
423 ? アイテム ↵
424 ? アイテム ↵
425 ? アイテム ↵
426 DEF ITEM ↵
427 VAR I, IX=CALLIDX ↵
428 ? --- しほう? ↵
429 IF SPVAR(IX, _HP)==0 THEN ↵
430 IF SPCHK(IX) AND #CHKS THEN RETURN ↵
431 ? --- けぞ ↵
432 SPCHR IX OUT I ↵
433 ADDSCORE (I-2048)+1 ↵
434 BEEP 12 ↵
435 SPCLR IX ↵
436 RETURN ↵
437 ENDIF ↵
438 ? --- プレイヤー/プレイヤーのママが空になった? ↵
439 VAR DD=SPHITSP(IX, PLTOP, PSEND) ↵
440 ? --- いどう ↵
441 VAR MV=MOV OBJ(IX, 1, 0) ↵
442 ? --- しほう? ↵
443 IF DD > -1 || MV == -1 THEN ↵
444 SPANIM IX, "S", -15, 2, 2, 1 ↵
445 SPANIM IX, "XY+", -15, 0, -32, 1 ↵
446 SPVAR IX, _HP, 0 ↵
447 ENDIF ↵
448 END ↵
449 ↵
450 ? スコアがさん ↵
451 ? スコアがさん ↵
452 ? スコアがさん ↵
453 ? VL: たずあたい ↵
454 ? ↵
455 DEF ADDSCORE VL ↵
456 SC=SC+VL ↵
457 IF HI < SC THEN HI=SC ↵
458 END ↵
459 ↵
```

アイテムの表示。

消えるときのアニメが終わるまで待つ。

プレイヤーかプレイヤーの弾が触ったら消える。

消えるときは拡大しながら上昇する。

スコア加算。

X1 と S1 ~ S5 のページを入力しないと敵やアイテムは表示されません

動くかどうか試したいときは、STARTボタンを押してみましょう

E2

■演出文字：文字列の解析

SETTEXT



```
460 *
461 * スプライトによるもじまじゅうじ
462 *
463 * □TX#: もじまじゅうじ
464 * □-----
465 * □MS: もじまじゅうじ
466 *
467 DEF □SETTEXT □B, TX#
468 VAR □I, L, N, T, X, Y, IX
469 * ---- □じゅんび
470 BEEP □B
471 L=LEN(TX#)
472 X=(400/2)-(L*16/2)+8
473 Y=(240/2)-(16/2)+8
474 * ---- □もじまじゅうじぶんくりがえす
475 MS=0
476 FOR □I=0 TO □L-1
477 * ---- □1もじまじゅうじせいせい
478 □N=ASC(MID$(TX#, I, 1))
479 □IF □N<&H20 □&&□N>&H5F THEN □N=342
480 □IF □N>&H20 THEN
481 □□' ---- □あきしゅどく
482 □□IX=NEWOBJ(□TXTOP, TXCNT, N, "TEXT"□)
483 □□IF IX== -1 THEN BREAK
484 □□' ---- □アニメーション
485 □□T=- (16+I*8)
486 □□SETOBJ □IX, X, Y, 0, 0
487 □□SPHOME □IX, 8, 8
488 □□SPCOLOR □IX, RGB(0, 255, 255, 255)
489 □□SPANIM □IX, "C", T, RGB(240, 255, 255, 255), 1
490 □□SPSCALE □IX, 4, 4
491 □□SPANIM □IX, "S", T, 1, 1, -(L-I)*16, 1, 1, T, 0, 0, 1
492 □□' ----
493 □□MS=MS+1
494 □□ENDIF
495 □□X=X+16
496 NEXT
497 END
498
```

与えられた文字列をスプライトとして表示。

対応文字は、アルファベットの大文字と数字。

左から1文字ずつでて全て表示されたら右から消えるアニメーションを登録。。

M1

■演出文字：表示中アニメ終了確認

TEXT



```
499 * [ ]
500 * [スプライトもじかんり]
501 * [ ]
502 DEF TEXT
503 VAR IX=CALLIDX
504 * --- アニメの終わりをかくにん
505 IF SPCHK(IX)==0 THEN
506   SPCLR IX
507   MS=MS-1
508 ENDIF
509 END
510
```

スプライトによる文字表示のアニメーションの終わりを待つ。



M1 ~ M2 のページを入力すると動きのある文字が表示されます

動くかどうか試したいときは、STARTボタンを押してみましょう

M2

■マップ表示：ステージ1と2のデータ

MAKESTAGE



```

511 '
512 ' [ステージせいせい]
513 '
514 ' □S:ステージばんごう
515 '
516 DEF MAKESTAGE S
517 '
518 @STG0
519 DATA " " '0
520 DATA " " '1
521 DATA " " '2
522 DATA " " '3
523 DATA " " '4
524 DATA " I P " '5
525 DATA " I I E E " '6
526 DATA " " '7
527 DATA " " '8
528 DATA " " '9
529 DATA " E " '10
530 DATA " " '11
531 DATA " " '12
532 '
533 @STG1
534 DATA " " '0
535 DATA " " '1
536 DATA " " '2
537 DATA " " '3
538 DATA " " '4
539 DATA " P I E E E I " '5
540 DATA " " '6
541 DATA " " '7
542 DATA " I I I E I E " '8
543 DATA " " '9
544 DATA " " '10
545 DATA " I E I E " '11
546 DATA " " '12
547 '
    
```

ステージごとのマップデータ。

ステージ1と2。

P=プレイヤー、E=敵、I=アイテム



S1

■マップ表示：ステージ3のデータ

@STG2



```
548 @STG2↵
549 DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 0↵
550 DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 1↵
551 DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 2↵
552 DATA " P □□ I I I □□□□□□□□□□□□□□□□□□□□□□ " ' 3↵
553 DATA " ■■■■ □□ □□ E E E E E E I " ' 4↵
554 DATA " ■■■■ □□ □□ E E E E E E E E " ' 5↵
555 DATA " ■■■■ □□ □□ E E E E E E E E " ' 6↵
556 DATA " ■■■■ □□ □□ E E E E E E E E " ' 7↵
557 DATA " ■■■■ □□ □□ E E E E E E E E " ' 8↵
558 DATA " ■■■■ □□ I E E E I □□□□□□□□□□□□ " ' 9↵
559 DATA " ■■■■ □□ E E E E I □□□□□□□□□□□□ " ' 10↵
560 DATA " ■■■■ □□□□ □□□□ □□□□ E □□□□□□□□□□□□ " ' 11↵
561 DATA " ■■■■ □□□□ □□□□ □□□□ □□□□ □□□□ □□□□ " ' 12↵
562 ↵
```

ステージ3のマップデータ。



★ヒント

マップを追加する時は、@STG+数字のラベルとDATAを増やします。
例えば、ステージ4を増やす場合・・・

```
@STG4↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 0↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 1↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 2↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 3↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 4↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 5↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 6↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 7↵
DATA " □□□□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 8↵
DATA " □□ P □□□□□□□□□□□□□□□□□□□□□□□□□□ " ' 9↵
DATA " □□□□ □□□□ □□□□ I I I □□□□ □□□□ E □□□□□□□□□□□□ " ' 10↵
DATA " □□□□ □□□□ □□□□ □□□□ □□□□ □□□□ □□□□ " ' 11↵
DATA " □□□□ □□□□ □□□□ □□□□ □□□□ □□□□ □□□□ " ' 12↵
```

■マップ表示：マップ用文字列の解析

@STG99



```
563 @STG99
564
565 VAR C, X, Y, IX, VX, OY=16*2
566 VAR C$, M$="@STG"+FORMAT$( "%D", S)
567 EC=0
568 ' --- せんざいする?
569 IF CHKLABEL(M$)==FALSE THEN
570   MAKEMAPS
571   RETURN
572 ENDIF
573 ' --- マップがいせきとひょうじ
574 RESTORE M$
575 FOR Y=0 TO 12
576   READ M$
577   ' --- もしれつぶんせき
578   FOR X=0 TO LEN(M$)-1
579     C$=MID$(M$, X, 1)
580     ' --- プレイヤー?
581     IF C$=="P" THEN
582       MKPLAYER X*16+8, OY+Y*16
583       C$=""
584     ENDIF
585     ' --- テキ?
586     IF C$=="E" THEN
587       MKENEMY X*16+8, OY+Y*16, (RND(3)-1)*0.1
588       C$=""
589     ENDIF
590     ' --- アイテム?
591     IF C$=="I" THEN
592       MKITEM X*16+8, OY+Y*16, RND(16)+2048
593       C$=""
594     ENDIF
595     ' --- もしれつおきかえ
596     M$=SUBST$(M$, X, 1, C$)
597   NEXT
598 ' --- マップ
599 C=RGB(0, (15-Y)*16+8, Y*4)
600 GPUTCHR 0, OY+Y*16, M$, 2, 2, C
601 NEXT
602 END
603
```

ステージごとの文字列を使ったマップデータから実際の画面にグラフィックを使って表示する。

マップ文字列を分析。

Pだったらプレイヤーを生成。

Eだったら敵を生成。

Iだったらアイテムを生成。

使われなかった文字列をマップとして表示。

★マップから敵を作る

S3

■自動マップ作成：グラフィック画面への表示

MAKEMAP



```
604 *
605 * ランダムなちけいせい |
606 *
607 DEF MAKEMAP S
608 VAR X, Y, C=RGB(0, 160, 0)
609 VAR MY=64
610 VAR OX=0, OY=MY+RND(150)
611 * ---
612 WHILE OX<400
613 OX=OX+RND(20)+4
614 OY=OY+(RND(100)-50)
615 IF OY<MY THEN OY=MY
616 IF OY>220 THEN OY=220
617 GLINE OX, OY, X, Y, C
618 OX=X:OY=Y
619 WEND
620 GPAINT 0, 239, C
621 * ---
622 MKPLAYER RND(100)+8, 0
623 * ---
624 FOR C=0 TO S+2
625 MKENEMY RND(300)+100, 0, (RND(3)-1)*0.1
626 NEXT
627 * ---
628 FOR C=0 TO S+1
629 MKITEM RND(300)+50, RND(100), RND(16)+2048
630 NEXT
631 END
632 *
```

マップデータが存在しないステージはプログラムでマップデータを自動的に生成する。

プレイヤー、敵、アイテムなどもステージ数に合わせて適度に調整しながら自動生成する。

※まれにプレイヤーと敵が同じ位置に出ることあり



S4

■自動マップ作成：プレイヤー等の生成

MKPLAYER



```
633 *
634 * |プレイヤーはっせい|
635 *
636 * □X, Y: はっせいほしよ
637 *
638 DEF□MKPLAYER□X, Y
639 VAR□IX=NEWOBJ(□PLTOP, PLCNT, PLDEF, "PLAYER"□)
640 IF□IX== -1 THEN RETURN
641 SETOBJ□IX, □X, Y, □0, □0
642 SPANIM□IX, "I", -8, PLDEF, -8, PLDEF+1, 0
643 END
644
645 *
646 * |タキはっせい|
647 *
648 * □X, Y: はっせいほしよ
649 * □S: スピード
650 *
651 DEF□MKENEMY□X, Y, S
652 VAR□IX=NEWOBJ(□ENTOP, ENCNT, ENDEF, "ENEMY"□)
653 IF□IX== -1 THEN RETURN
654 SETOBJ□IX, □X, Y, S, □0
655 SPCOLOR□IX, #GREEN
656 ANIMOBJ□IX, ENDEF, S
657 EC=EC+1
658 END
659
660 *
661 * |アイテムはっせい|
662 *
663 * □X, Y: はっせいほしよ
664 * □C: ほんごう
665 *
666 DEF□MKITEM□X, Y, C
667 IX=NEWOBJ(□ITTOP, ITCNT, C, "ITEM"□)
668 IF□IX== -1 THEN RETURN
669 SETOBJ□IX, □X, Y, □0, □0
670 END
671 *
```

適当にプレイヤーを生成。

適当に敵を生成。

★敵スプライトを作る

適当にアイテムを生成。

動くかどうか試したいときは、STARTボタンを押してみましょう



■共通：スプライト新規作成と情報取得

NEWOBJ



```
672 *
673 * しんきスプライトさくせい
674 *
675 * □T:さいしょのほんごう
676 * □C:さいごいごさう
677 * □N:SPDEFIほんごう
678 * □FUNC#:わりあてるプログラム
679 *
680 DEF□NEWOBJ(□T, C, N, FUNC#□)
681 VAR□J, IX
682 SPSET□T, T+C-1, N□OUT□IX
683 IF□IX>-1□THEN
684 □SPFUNC□IX, FUNC#
685 □FOR□J=0□TO□7
686 □□SPVAR□IX, J, 0
687 □NEXT
688 □SPVAR□IX, _HP, 1'どりあえず1いれておく
689 □SPCOL□IX, TRUE
690 ENDF
691 RETURN□IX
692 END
693
694 *
695 * スプライトきほんじょうほうしゅどく
696 *
697 * □IX:ほんごう
698 * □-----
699 * □X, Y:ぎひょう
700 * □VX, VY:いどうりょう
701 *
702 DEF□GETOBJ□IX□OUT□X, Y, VX, VY
703 SPOFS□IX□OUT□X, Y
704 VX=SPVAR(IX, _VX)
705 VY=SPVAR(IX, _VY)
706 END
707 *
```

与えられた番号と最大個数などの情報からスプライトに空きを探して新しいスプライトを生成する。

指定されたスプライト番号に関連づいた情報を取得。

■共通：スプライト情報保存とアニメ設定

SETOBJ



```
708 *
709 * スプライトまほんじょうほうほせん
710 *
711 * □IX:ほんごう
712 * □X, Y:ざひょう
713 * □VX, VY:いどうりょう
714 *
715 DEF□SETOBJ□IX, X, Y, VX, VY
716 SPOFS□IX, X, Y
717 SPVAR□IX, _VX, VX□'いどうりょうX
718 SPVAR□IX, _VY, VY□'いどうりょうY
719 END
720
721 *
722 * スプライトアニメせつてい
723 *
724 * □IX:ほんごう
725 * □N:まじゅんほんごう
726 * □VX, VY:いどうりょう
727 *
728 DEF□ANIMOBJ□IX, N, VX
729 VAR□W=-8
730 IF□VX<0□THEN□N=N+8
731 SPCHR□IX, N
732 SPANIM□IX, "I", W, N, W, N+1, W, N+2, W, N+3, 0
733 END
734
```

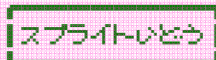

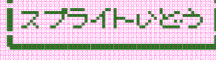










































与えられたスプライト番号の情報を記録。
スプライト内部変数領域に保管される。

指定されたスプライトにアニメーションを設定する。

■共通：スプライトの移動

MOV OBJ



```
735 *  
736 *  
737 * 
738 * □IX:ぼんご 
739 * □SPD:スピード 
740 * □GR:TRUE=あしもとかくにん 
741 * □----- 
742 * □-1のどましぼう 
743 * 
744 DEF □MOV OBJ(□IX, SPD, GR) 
745 VAR □I, R, X, Y 
746 * ---- 
747 GET OBJ □IX □OUT □X, Y, VX, VY 
748 * ---- 
749 IF □GSPDIT(X, Y+VY+1)==0 □THEN 
750 □VY=VY+1 
751 □IF □VY>8 □THEN □VY=8 
752 ELSE 
753 □VY=0 
754 ENDIF 
755 * ---- 
756 I=VX*SPD 
757 * ---- 
758 IF □GR □&& □GETCOL(X+I, Y+1)==0 □THEN □I=0 
759 * ---- 
760 IF □MS>0 □THEN □I=0 
761 * ---- 
762 IF □I!=0 □&& □GETCOL(X+I, Y)!=0 □THEN 
763 □? ---- 
764 □I=0 
765 □IF □GETCOL(X, Y-1)==0 □THEN □VY=-1 
766 ENDIF 
767 * ---- 
768 X=X+I 
769 IF □X<0 □□□ THEN □X=0 □: VX=-VX 
770 IF □X>399 □THEN □X=399 □: VX=-VX 
771 Y=Y+VY 
772 * ---- 
773 IF □Y>240+16 □THEN □RETURN □-1 
774 * ---- 
775 SET OBJ □IX, □X, Y, □VX, VY 
776 RETURN □IX 
777 END 
```

指定されたスプライト番号とスピードなどの情報から、足元を調べて歩く。

足元に何もなければ落下するために加速。

GRがTRUEの時だけ進む前に足元確認。

動いた結果、壁だったら上に進む。

左右の限界値を超えそうになったら移動方向反転。

画面外に落ちたら死亡。


動くかどうか試したいときは、STARTボタンを押してみましょう

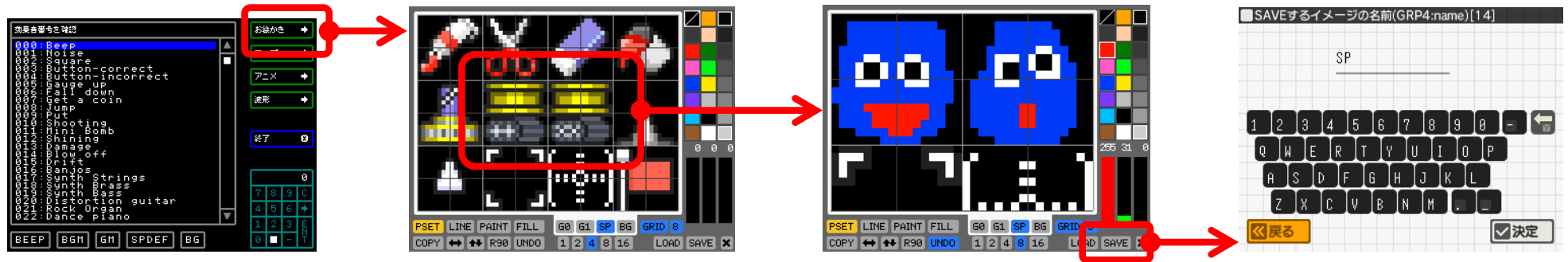
X1

時間があったら試してみよう

おまけ

■自分で描いた絵に変更してみよう！

- お絵かきツールで自分の砲台？を描き替えて保存（SAVE）
 -  スマイルボタンからスマイルツールを起動して「お絵かき」のボタンを押します



- プレイヤーの砲台キャラクターの位置を好きな絵に書き替えます
- SAVEボタンから「SP」という名前で保存します
- ツール右下の「X」ボタンでツールを終了します

※ここで画面がゴチャゴチャしていたら **ACLS** を実行してください

- プログラムからの読み込み（LOAD命令）

```
52 ' ---□フォントへんこう↓  
53 ' LOAD□" GRPF:FONT", FALSE↓  
54 LOAD " GRP4:SP", FALSE↓  
55 ' --- BGではいけいをつくる↓  
FOR□X=0□TO□31↓
```

LOAD命令を1行追加してからSTARTボタンを押してみましよう。
ツールで作った自分のキャラクターが表示されるはずです。



■自分で作った音楽を鳴らしてみよう！

- ・ 譜面（スコア）と音の高さ（音程）に割り当てられた記号

音の高さ	C	D	E	F	G	A	B	C	A	
日本での読み	ド	レ	ミ	ファ	ソ	ラ	シ	ド	ラ	

- ・ 適当な行に以下の音楽演奏データとプログラムを追加します

```

54 LOAD" GRP4:SP", FALSE
55 ? --- MYBGM
56 @BGM1
57 DATA":0T100"
58 DATA":1@1 L4 04[ CDEFGAB<C]"
59 DATA":2@129L1602[[G-A-E-B-]4]"
60 DATA":3@129L1602[ CRRRERRRRCRCRERRE-]"
61 DATA":4@38 L1602[[C<C]8]"
62 DATA0
63 BGMSETD 128, @BGM1
64 ? --- BGではいけしをつくる
    
```

- ・ メインループの近くに 1 行追加

```

84 ? 〇メインループ〇
85 ? =====
86 VARGM=1
87 BGMPLAY〇128
    
```

★ヒント

演奏を止める時は、 **BGMSTOP**

<使われているMMLの説明>

@1はピアノ、@129はドラム、@38はシンセベース
 L4、L8、L16は音の長さ、[]は、ループ
 02、04はオクターブ、<>でオクターブの上げ下げ

詳しくは、MMLでインラインヘルプを見てみましょう

■そのほかの改造アイデア

- 敵の改造や種類を増やしてみよう！
 - 歩いている方向にだけ弾を発射させるには？
 - もっと素早く動かすには？

- 敵の種類を増やしてみよう！



- 上記3ページの中に敵（緑ミイラ）に必要なプログラムが含まれています
 - 必要な範囲を丸々コピーして貼り付けてからENEMYという名前をENEMY2とかに変更すれば新しい敵の完成
 - マップ上の配置は「E」じゃない文字で新しい敵を指定できるようにしましょう（例えば「X」「Y」「Z」とか）
- 空をふわふわ飛びまわって時々フンを落とす鳥とか・・・
- 緑色に塗りつぶすタイプの爆弾を出す敵・・・
- アイテムを吐き出しまくる敵？
- プレイヤーの攻撃方法を変化させてみよう
 - アイテムを取ったらスピードアップ？
- 制限時間をつけてみる？
 - 時間内にクリアしないとGAMEOVER

ゲームを作っている人達の仕事内容

ゲームクリエイターになるためには？

■ゲームの仕事とクリエイターに求められる要素

ゲームの開発は一人ではなくグループで作ることが多くなってきました。

- 作る人の仕事の役割が二つに分かれてきています（クリエイターとワーカー）
- どちらも重要な仕事ですが生み出す楽しみを楽しみたい人はクリエイターがお勧め
- もちろんプログラマ、デザイナー、プランナーなどの仕事の種類での分類もあります

<クリエイター（新しいことを次々生み出す人）>

- 何もないところからアイデアを生み出します（ゲームのルール、画面のデザイン、新しいプログラム）
- 作りたい人の話を聞いて面白いアイデアを次々と提案します
- 未来のお客様を想像して迷わせない操作方法を考えます

<ワーカー（指示された内容を忠実に作りこむ人）>

- もくもくと指示された作業を進めます（仕様書や指示書などに基づいてプログラムやデザインを行う）
- 大量のデータやプログラムを作るために必要なメンバーです

クリエイターを目指すのであれば・・・

- 面白いことは積極的にチャレンジしてみる（遊ぶことも、勉強も面白いと思ったらトコトン楽しむ）
- ダジャレを考える（常に何か他のものに置き換える訓練としてダジャレは有効だと思います）
- 遊んでくれる人のことを想像しながらアイデアを考える（バーちゃんでも遊べるかな？とか）

ぜひ、未来のゲーム開発者として面白いゲームを生み出す人になってください!!