

Guida pratica alla creazione dei sistemi

Il linguaggio di Xtreme è un insieme di azioni e condizioni che vi permettono di giocare il sistema creato in automatico, piazzando i pezzi sul tappeto e registrando l'andamento delle puntate.

Ad ogni spin (boule) viene eseguito il sistema seguendo l'ordine dei comandi inseriti

Il linguaggio è simile alla lingua inglese e la maniera migliore per programmare un sistema in Xtreme è verbalizzare scrivendo su un foglio tutte le azioni che compiereste al tavolo per effettuare la puntate

Esempio:

- Registrare le ultime tre chance rosso/nero [track last 3 red/black]
- Aspettare che escano tre rossi di fila [if record "rossi" layout =3]
- Giocare 1 pezzo sul rosso [put 1 pezzo sul red]
- Se si vince fermarsi e aspettare che escano ancora tre rossi di fila [If Net>0]
- Se si perde giocare ancora rosso ma con 2 pezzi [If Net<0 Put 2 pezzi sul red]

I comandi sono semplici, bisogna solo imparare un po' di logica e le varie relazioni tra loro. Se i sistemi non sono ricchi di calcoli astrusi o complessi, la parte più difficile non sarà programmare il sistema in xtreme ma verbalizzarlo in maniera dettagliata per iscritto

- Method= Un blocco di comandi (il method Main è il principale senza il quale il programma non gira)
- Action= sono dei comandi che permettono di compiere delle azioni di diverso tipo
- Condition= sono delle condizioni che si mettono all'interno del sistema per scegliere di effettuare determinate azioni
- Record Data = sono dei piccoli "cassettini" in cui si depositano dei dati numerici.
- Record Layout = altri cassettoni in cui si depositano delle informazioni sulle combinazioni della roulette (rosso, dozzina 1, etc)
- Record...Index= I cassettetti possono essere singoli o delle vere e proprie cassettiere, in questo caso l'index è la numerazione progressiva dei cassettetti all'interno della cassettiere.
- Flag = sono dei cassettetti singoli in cui si possono mettere solo due tipi di informazioni: vero o falso e servono per la verifica delle condizioni.

- A tutte le possibili posizioni di fiches sul tavolo della roulette corrisponde un layout che si identifica con il nome della puntata in inglese.
- Esempi: Number 5, Split 24:27 (cavallo), Line 1-6 (sestina, etc)
- I layout rivestono un ruolo fondamentale in Xtreme perché corrispondono al tavolo della roulette, quindi potete usarli per mettere o togliere le fiches, per vedere se qualche combinazione ha vinto o perso.
- Inoltre oltre ad identificare le posizioni sul tavolo servono anche per tenere traccia delle combinazioni uscite e verificare l'avverarsi o meno di alcune condizioni, pensiamo alle figure da tre (rosso, nero, nero) o alla ripetizione consecutiva delle combinazioni, o ai ritardi.
- Anche per i numeri sulla ruota si possono facilmente usare dei particolari comandi (distanza, posizione, settori) per verificare ed eseguire delle analisi.
- Una delle particolarità di Xtreme è che ad ogni boule (spin) le fiche vengono realmente piazzate sul tavolo e quindi il risultato della puntata viene utilizzato dal programma come se l'avessimo inserito manualmente. In questo modo si hanno a disposizione tutte le analisi e le statistiche ottenibili con Xtreme

- Abbiamo detto che i record sono come delle cassettiere. Queste liste possono essere utili per diversi motivi, il più comune è la progressione delle puntate. Basta mettere il numero di pezzi che si vuole puntare ad ogni termine della progressione e poi muoversi in su e in giù per restituire alla puntata il valore desiderato.
- Nel caso dei layout le liste sono comode per memorizzare dei gruppi di puntate come le finali, le dozzine e le colonne associate, settori della ruota occupati con pieni e cavalli, etc. Basterà richiamare la lista per nome per poi puntare dei pezzi o verificare delle condizioni.
- Chi utilizza schemi tipo i quadrati di Zantilfiore o altri raggruppamenti potrà semplicemente riprodurli tramite una serie di liste.

- I record di dati servono per effettuare delle operazioni numeriche. Attualmente Xtreme è poco flessibile per quanto riguarda la gestione delle operazioni, miglioreranno anche questo.
- 100% questa parte di comando associata solitamente alle operazioni base e al comando put (metti) viene utilizzata per assegnare dei valori da un record all'altro. Inoltre si utilizza per effettuare delle operazioni sulle fiche puntate ad esempio un paroli (put 100% win – Metti il 100% della vincita su) o la suddivisione della puntata 50%
- I dati vengono usati anche come contatori per tenere traccia del numero di puntate, dell'attacco, dei colpi di attesa, del numero di vittorie e sconfitte.
- La finestra Statistiche>Record dati di Xtreme vi permette di visualizzare l'andamento di tutti i record ad ogni boule. Utile per inserire dei parametri di controllo statistici da visualizzare quando si gioca on-line con Xtreme attivo (ad esempio lo scarto o l'ultima figura da tre)
- Il Bankroll è una variabile che identifica il saldo di cassa e che si aggiorna automaticamente ad ogni puntata. Viene utilizzata nelle condizioni per verificare se si è raggiunto l'obiettivo di vincita o il massimo passivo sostenibile.

- Confondere i record tra layout e dati o dimenticarsi di specificarli
- Dimenticarsi di inserire un contatore che aumenti all'interno del loop
- Non re-indicizzare le liste prima di leggere o scrivere in un record
- Verificare che ai numeri con più layout uguali venga associato quello desiderato (ad esempio il 7 fa parte sia della sestina 4/9 che della 6/12)
- Richiamare le subroutine in ordine sbagliato o non chiamarle perché all'interno di altre subroutine soggette a condizioni
- Il sistema considera perse tutte le puntate piazzate salvo poi restituire le puntate messe sulle combinazioni vincenti (l'errore capita quando si usa l'azione loss per identificare le puntate perse)
- Dimenticarsi di aprire o chiudere una condizione con begin e/o end

Prendiamo come esempio il sistema “Terzine in calore” presente sul sito
Iniziamo con il verbalizzare il sistema

1. Registrare 18 terzine consecutive
2. Vedere quale terzina è uscita di più
3. Verificare se si verifica una delle seguenti condizioni
 1. se c'è più di una terzina con lo stesso valore
 2. se la terzina in calore supera le 4 uscite
4. Se le condizioni sono tutte false mettere in gioco la terzina in calore
5. Puntare 1 pezzo per 12 boules, dalla 13 alla 18 puntare 2 pezzi
6. In caso di vincita, o di sconfitta alla 18esima boule
 1. Fermarsi
 2. Controllare le ultime 18 terzine e riprendere dal punto 2
7. Fermarsi con uno stopwin e uno stoploss a scelta

0 – Preparazione del codice base

```
system "Terzine in calore"  
{  
  il mio primo sistema  
}  
method "main"  
begin  
  
end
```

Per iniziare scegliere “nuovo sistema” apparirà di default un blocco di codice precompilato.

Cambiate il nome nelle virgolette dopo system

Scrivete una breve descrizione o dei vostri appunti nelle righe tra le parentesi graffe

Il metodo “main” non va modificato perché è la routine principale senza la quale il sistema non funziona.

All’interno dello spazio delimitato da begin e end dovrete scrivere i comandi principali

1 – Inizializzare la sessione

```
method "main"  
begin  
    while Starting a new Session  
    begin  
  
    end  
end
```

La prima cosa da fare è inserire una condizione che verifica se ci si trova all’inizio di una nuova sessione.

Questa condizione è utile per inizializzare i record ed i flag o per chiedere all’utente alcuni parametri da utilizzare (ad esempio se la roulette è di tipo francese o americana).

Per praticità e pulizia del codice i comandi di input (richiesta di feedback all’utente) e le inizializzazioni se sono troppo lunghe, vengono raccolti in una routine a parte e richiamati con un comando “call”

2 – Inizializzare la sessione – Richiesta info utente

```
while Starting a new Session  
begin  
    call "Richieste"  
end
```

```
method "Richieste"  
begin  
    group  
        begin  
            input data "Fermarsi in vincita di " record "SW" Data  
            input data "Fermarsi quando si è sotto di " record "SL" Data  
        end  
    end  
end
```



Inserire un comando call per richiamare la subroutine, che non è altro che un nuovo blocco method a cui si assegna il nome prescelto, in questo caso “Richiesta”.

All’interno della routine richiesta si trova il blocco Group che viene utilizzato per inserire più comandi all’interno della stessa finestra di dialogo, senza il group si avrebbero tante finestre che si aprono ad ogni comando input inserito.

Con i comandi input data assegniamo ai record SW e SL i valori che inserirà l’utente per poi usarli all’interno del sistema. In alto a destra potete vedere come questa finestra apparirà durante l’esecuzione.

C’è chi preferisce impostare la cassa “Bankroll” e segnalare lo stoploss quando la cassa è a zero

3 – Inizializzare la sessione e definire la progressione

```
set list [1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2] record "Progressione" data
copy list [Street(1-3),Street(4-6),Street(7-9),Street(10-12),Street(13-15),
          Street(16-18),Street(19-21),Street(22-24),Street(25-27),Street(28-30),
          Street(31-33),Street(34-36)] nel record "Terzine" layout
set flag "Gioca" false
```

La progressione in questo sistema è molto semplice, in effetti si potrebbe anche gestire con una condizione di controllo (dopo la 12esima boule punta 2 pezzi).

E' preferibile però utilizzare i record con più elementi perché è più semplice conteggiare le puntate riferendosi all'indice ed in caso di modifica del sistema, se si vuole allungare o modificare i termini della progressione, basterà cambiare solo i valori nella lista.

Creiamo anche una lista contenente tutte le terzine (street), in questo modo indicizziamo la terzina con un numero progressivo. Record che ci sarà utile in seguito in fase di elaborazione.

Come vedete in questo caso, dato che si tratta di layout, utilizziamo il comando Copy list e non Set list

Abbiamo anche aggiunto un flag "gioca" settato su false, che utilizzeremo per dire al sistema che si può puntare; con il flag settato su false le boules diventano colpi di attesa

Qui si può notare la libertà di formattazione del linguaggio, si può andare a capo e spostarsi a piacere

4 – la prima boule

```
(esempio A)  while Starting a new Session
              begin
                call "Richieste"
                set list [1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2] record "Progressione" data
                set flag "Gioca" false
                exit
              end

              while on Each Spin
(esempio B)  begin

              end
```

Abbiamo detto che tutto quello che avviene all'interno del blocco main viene eseguito ad ogni boule. Quindi qualunque sia il sistema che andiamo a sviluppare al termine della condizione di “verifica se nuova sessione” abbiamo due possibilità o iniziare subito a puntare al tavolo o attendere un colpo di attesa.

Per iniziare a giocare al tavolo basta inserire semplicemente il codice all'interno del blocco main. Se si vuole attendere una boule ci sono due possibilità o inserire il comando exit all'interno della condizione (esempio A) oppure utilizzare la condizione While on each spin che significa: ad ogni boule esegui le istruzioni all'interno del blocco (esempio B). Noi utilizzeremo il B per rendere più leggibile il codice

5 – Colpi di attesa e registrazione boules

```
track last street 18 nel record "18 terzine" layout
if record "18 terzine" layout count <18
begin
    exit
End
call "Verifica"
```

Il sistema prevede la registrazione dell'uscita di 18 terzine, prima di poter eseguire qualsiasi altra azione.

Con il comando "track last" memorizziamo le uscite progressivamente (a mano a mano che escono) nel record "18 terzine". Come potete vedere questo record è di tipo layout, perché andiamo a memorizzare una combinazione della roulette non un valore numerico.

Il blocco che segue è una condizione **if** semplice che controlla **se** il numero di terzine all'interno del record è inferiore a 18 (il comando count conta quanti sono gli elementi all'interno del record) allora interrompe l'esecuzione e passa subito ad una nuova boule con il comando exit.

Se si arriva a 18 invece il blocco if risulta falso quindi l'esecuzione del programma passa alla riga successiva dove abbiamo inserito una chiamata alla subroutine "Verifica"

6 – Routine di controllo delle terzine

```
track last street 18 nel record "18 terzine" layout
if record "18 terzine" layout count <18
begin
    exit
End
call "Verifica"
```

Il sistema prevede la registrazione dell'uscita di 18 terzine, prima di poter eseguire qualsiasi altra azione.

Con il comando "track last" memorizziamo le uscite progressivamente (a mano a mano che escono) nel record "18 terzine". Come potete vedere questo record è di tipo layout, perché andiamo a memorizzare una combinazione della roulette non un valore numerico.

Il blocco che segue è una condizione **if** semplice che controlla **se** il numero di terzine all'interno del record è inferiore a 18 (il comando count conta quanti sono gli elementi all'interno del record) allora interrompe l'esecuzione e passa subito ad una nuova boule con il comando exit.

Se si arriva a 18 invece il blocco if risulta falso quindi l'esecuzione del programma passa alla riga successiva dove abbiamo inserito una chiamata alla subroutine "Verifica"

7 – Routine di Verifica – le operazioni da effettuare

```
method "Verifica"  
begin  
  if flag "Gioca" true  
    begin  
      return  
    end  
  end  
end
```

Qui si entra nel cuore del sistema, ovvero i comandi che analizzano le terzine uscite e fanno i controlli. Utilizzeremo due routine di ciclo, ovvero delle routine che permettono di eseguire ripetutamente una serie di comandi finché la condizione non viene rispettata.

Innanzitutto il codice sopra va inserito per comunicare al programma che se si sta puntando non occorre elaborare alcuna verifica. Se il flag è true interrompe subito la routine.

Per ogni terzina partendo dalla 1 alla 12 (ciclo principale) leggeremo tutte le 18 terzine uscite (ciclo secondario) per vedere se quella terzina è uscita, se è uscita la aggiungiamo ad un record che chiameremo “conta nelle 18”, in questo record avremo il conteggio di quante volte le singole terzine sono uscite. Inoltre ad ogni terzina faremo un controllo per vedere se è quella con più uscite, per trovare il massimo delle uscite nelle 18 boules.

Il programma quindi eseguirà per 12 volte (le terzine) un ciclo da 18 (le terzine uscite).

Terminati cicli avremo un record contenente per ogni terzina il totale di quante volte è uscita, un record con il numero massimo di uscite di una o più terzine. Con questi elementi potremo verificare le condizioni del sistema, ovvero verificare se il massimo delle uscite è superiore a 4 e quante sono le terzine in calore.

8 – Routine di Verifica – il loop principale

```
put 1 record "Terzine" layout index
loop until record "Terzine" layout index > record "Terzine" layout count
begin
    ...
    ...
    add 1 record "Terzine" layout index
end
```

Sopra potete vedere la routine di ciclo base, i ... stanno ad indicare lo spazio per i comandi da fare eseguire ad ogni ciclo.

Il loop deve avere come minimo i seguenti comandi:

La prima riga setta a 1 l'indice del record "Terzine"

La seconda è la condizione da soddisfare ovvero confronta l'indice con il totale dei record. se l'indice supera il conteggio il ciclo si interrompe (la condizione diventa vera quando si raggiunge la fine del record)

La riga con Add aumenta di 1 l'indice

Quindi tutti i comandi compresi tra begin ed end verranno eseguiti tante volte quanti sono gli elementi nel record "Terzine"

9 – Routine di Verifica – il loop secondario

```
put 1 record "18 terzine" layout index
loop until record "18 terzine" layout index > record "18 terzine" layout count
begin
    ...
    ...
    add 1 record "18 terzine" layout index
end
```

All'interno del loop si possono inserire altre condizioni, nel nostro caso inseriremo un altro loop per poter confrontare due record indicizzati e cioè "Terzine" e "18 terzine".

Come potete vedere i due loop hanno la stessa struttura, quello che cambia è solo la loro posizione. Il loop secondario deve essere contenuto tutto all'interno del loop principale (ovvero tra begin ed end) altrimenti il programma segnalerà un errore.

10 – Routine di Verifica – le assegnazioni

```

if record "Terzine" layout = record "18 terzine" layout
begin
  add 1 record "Conta nelle 18" data
end
if record "Conta nelle 18" data > record "Massimo" data
begin
  put 100% record "Conta nelle 18" data nel record "Massimo" data
end

```

Il primo blocco if esegue la seguente operazione: Se il layout presente nel record terzine è uscito lo aggiunge all'elemento corrispondente nel record "conta nelle 18". Il secondo controlla se il totale delle volte in cui è uscita questa terzina è maggiore del massimo numero trovato fino a quel momento, nel caso lo fosse il totale di questa terzina diventa il nuovo valore massimo di riferimento (memorizzato nel record "Massimo").

Introduciamo qui un elemento di analisi molto importante, la finestra dei record dati che si trova nel menu "Statistiche>record dati". Qui vengono mostrati ad ogni boules i valori di tutti i record.

0		19	Street(10-12). Street(16-18). Street(25-27). Street(34-36). Street(25-27). Street(16-18). Street(10-12). Street(16-18). Street(22-24). Street(19-21). Street(13-15). Street(19-21). Street(28-30). Street(10-12). Street(7-9). Street(10-12). Street(19-21). Street(10-12)	18 terzine
12	0, 0, 1, 5, 1, 3, 3, 1, 2, 1, 0, 1	0		Conta nelle 18
1	5	0		Massimo

Si può vedere la sequenza delle ultime 18 terzine, il conteggio di quante volte si sono ripetute ed il massimo di ripetizioni

Questa finestra è disponibile sempre ed è uno strumento molto utile quando si gioca on-line, si possono inserire dei record che non hanno nulla a che vedere con le puntate ma che servono per analisi statistiche

11 – Routine di Verifica – il primo conteggio

```
put 1 record "Terzine" layout index
loop until record "Terzine" layout index > record "Terzine" layout count
begin
    put 100% record "Terzine" layout index nel record "Conta nelle 18" data index
    put 1 record "18 terzine" layout index
    loop until record "18 terzine" layout index > record "18 terzine" layout count
    begin
        if record "Terzine" layout = record "18 terzine" layout
        begin
            add 1 record "Conta nelle 18" data
        end
        if record "Conta nelle 18" data > record "Massimo" data
        begin
            put 100% record "Conta nelle 18" data nel record "Massimo" data
        end
        add 1 record "18 terzine" layout index
    end
    add 1 record "Terzine" layout index
end
```

Qui sopra il codice completo dei due cicli di controllo, in cui si può vedere l'assegnazione al record "conta nelle 18" lo stesso indice delle terzine. Operazione effettuata con il comando put 100% che in questo caso va interpretato come "Metti tutto il contenuto.. nel.."

12 – Routine di Verifica – il secondo conteggio

```
put 1 record "Conta nelle 18" data index
loop until record "Conta nelle 18" data index > record "Conta nelle 18" data count
begin
  if record "Conta nelle 18" data = record "Massimo" data
  begin
    add 1 record "Num. Terzine in calore" data
    put 100% record "Conta nelle 18" data index nel record "Terzine" layout index
    copy record "Terzine" layout nel record "Da Puntare" layout
  end
  add 1 record "Conta nelle 18" data index
end
```

Ottenuti i valori con i primi loop adesso possiamo contare quante sono le terzine in calore, ovvero a quante terzine corrisponde il numero massimo di uscite.

Si effettua un ciclo all'interno del record "conta nelle 18" per confrontare il numero di ripetizioni della terzina con il massimo delle ripetizioni (blocco if).

Se viene trovato il valore si aggiunge 1 al conteggio del record "Num. Terzine in calore"

Le due righe rosse servono per identificare la terzina in calore memorizzandola nel record "da puntare", per farlo si assegna l'indice numerico del loop al record "terzine" e si copia il layout della terzina

13 – Routine di Verifica – la verifica delle condizioni di gioco

```
if record "Massimo" data <=4
and record "Num. Terzine in calore" data =1
begin
    Set flag "gioca" true
end
```

A questo punto abbiamo in mano tutti gli elementi per poter verificare le condizioni per poter attivare il gioco ed iniziare a puntare sul tavolo.

Qui si può vedere l'utilizzo di un if composto da due condizioni, in questo caso devono essere vere entrambe (and) affinché i comandi all'interno del blocco vengano eseguiti.

Quindi se il numero massimo è inferiore o uguale a 4 (condizione 1 del sistema) o se c'è solo una terzina in calore (condizione 2 del sistema) si fa diventare vero (true) il flag "gioca"

14 – Routine di Verifica – inizializzazioni relative

```
clear record "Massimo" data  
Clear record "Num. Terzine in calore" data  
set list [0,0,0,0,0,0,0,0,0,0,0,0,0] record "Conta nelle 18" data  
set flag "gioca" false
```

Per evitare che rimangano dei valori nei record che appartengono alla verifica precedente, occorre inizializzare i record ed i flag utilizzati nella routine di verifica.

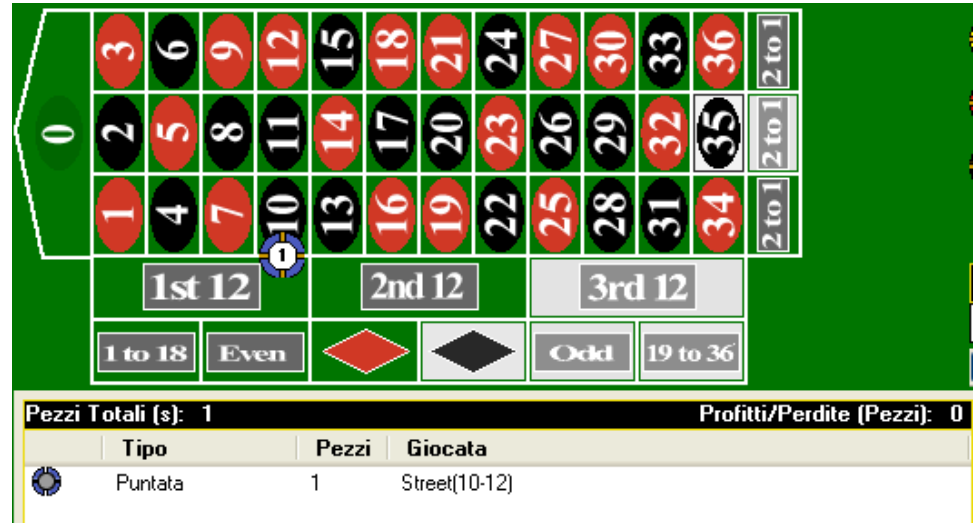
Questi comando vengono messi all'inizio della routine.

Abbiamo illustrato due modi per inizializzare i record o con il comando Clear (che elimina ogni contenuto del record) e come nel caso del record "conta nelle 18" mettendo a 0 tutti i valori (si mantiene l'indicizzazione)

15 – Puntate

```

call "verifica"
if flag "Gioca" true
begin
    put 100% record "Progressione" data
    sul record "Da Puntare" layout
end
    
```



Come abbiamo visto in precedenza se il flag “gioca” viene settato su true, significa che ci sono le condizioni per puntare, a questo punto, avendo già memorizzato nel record “da puntare” la terzina in calore, possiamo istruire il programma di metterla fisicamente a tappeto.

Ecco qui l'utilità di avere la progressione in una lista, con un solo comando che rimane sempre uguale, il sistema piazzerà i pezzi che gli verranno indicati in base al valore dell'indice della progressione.

Questo comando verrà inserito nel metodo Main dopo la chiamata alla routine di verifica.

16 – Risultati delle puntate - Metodo

```
method "Risultati"  
begin  
  if Any Street Bet won each  
  begin  
    ...  
  end  
  else  
  begin  
    ...  
  end  
end
```

Quello che manca per completare la puntata è l'indicazione di come comportarsi in base al risultato della giocata. Perciò creeremo un Method "Risultati" con la rispettiva chiamata di Call, che in questo caso verrà inserita all'inizio della boule (spin), prima della verifica, perché ovviamente (fatta eccezione per i colpi di attesa) è la prima operazione che si fa prima di iniziare a ripuntare.

Nella routine "Risultati" inseriremo una condizione IF associato all'azione Any Street bet won each, ovvero se "qualsiasi terzina ha appena vinto" a cui fa seguito un blocco di azioni da fare in caso di vittoria. La condizione ELSE (Altrimenti) che si associa ad un comando IF avrà il blocco alternativo che di conseguenza corrisponderà alle azioni da far ein caso di sconfitta.

17 – Risultati delle puntate – Se si vince

```
if Any Street Bet won each  
begin  
    set flag "Gioca" False  
    put 1 record "Progressione" data index  
end
```

Innanzitutto in caso di vittoria si setta il flag “Gioca” in false così si segnala al sistema che non deve più puntare, ma che dovrà iniziare nuovamente il controllo sulle ultime 18 boules

Poi dato che si tratta di una puntata con progressione, si risetterà a 1 il valore della progressione, così da poter iniziare la prossima puntata dal primo termine della progressione.

18 – Risultati delle puntate – Se si perde

```
else
begin
  add 1 record "Progressione" data index
  if record "Progressione" data index > record "Progressione" data count
  begin
    set flag "Gioca" False
    put 1 record "Progressione" data index
  end
end
end
```

La routine Else (Altrimenti) indica cosa far ein caso di sconfitta.

Per prima cosa salire di un termine nella progressione (muovendo l'indice)

Poi si controlla se si è arrivati in fondo alla progressione, in questo caso bisogna smettere di puntare settando il flag "gioca" su false e rimettendo a 1 l'indice della progressione.

Questa routine è present ein tutti i sistemi che usano le progressioni.

19 – Risultati delle puntate – Controllo saldo

```
if Bankroll >= record "SW" data  
or Bankroll <= record "SL" data  
begin  
    Stop Session  
end
```

Prima di chiudere la routine risultati è opportuno valutare la situazione di cassa alla luce degli ultimi eventi.

Si confronta il saldo (bankroll) con quelli che erano gli obiettivi fissati dall'utente all'inizio della sessione.

Più precisamente se il saldo è maggiore o uguale allo stopwin o minore o uguale allo stoploss si interrompe la sessione.

20 – Risultati delle puntate – Routine completa

```
method "Risultati"  
begin  
  if Any Street Bet won each  
    begin  
      set flag "Gioca" False  
      put 1 record "Progressione" data index  
    end  
  else  
    begin  
      add 1 record "Progressione" data index  
      if record "Progressione" data index > record "Progressione" data count  
        begin  
          set flag "Gioca" False  
          put 1 record "Progressione" data index  
        end  
      end  
    end  
  //Controlla stopwin e stoploss  
  if Bankroll >= record "SW" data  
  or Bankroll <= record "SL" data  
  begin  
    Stop Session  
  end  
end  
end
```

Utilità

```

Editor Sistemi [#Terzineincalore.dgt]
File Esegui Cerca Opzioni Help

1 |system "Terzine in calore"
2 |{
3 |  il mio primo sistema
4 |}
5 |method "main"
6 |begin
7 |  while Starting a new Session
8 |  begin
9 |    Call "Richieste"
10 |   Set list [1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2] record "P
11 |   copy list [Street(1-3),Street(4-6),Street(7-9),Street(10-
12 |   Street(16-18),Street(19-21),Street(22-24),Stre
13 |   Street(31-33),Street(34-36)] nel record "Terz
14 |   Set flag "Gioca" false
15 |   Put 1 record "Progressione" data index
16 |
17 |  end
18 |
19 |  While on Each Spin
20 |  begin
21 |    track last street 18 nel record "18 terzine" layout
22 |    if record "18 terzine" layout count <18
23 |    begin
24 |      exit
25 |    end
26 |    if flag "Gioca" true
27 |    begin

```

Boule	Numero	Tipo	Puntati	Vinti	Persi	Netto	Saldo Pezzi	Giocate
15	3	Nessuna puntata	0	0	0	0	0	
16	12	Nessuna puntata	0	0	0	0	0	
17	2	Nessuna puntata	0	0	0	0	0	
18	6	Nessuna puntata	0	0	0	0	0	
19	33	Nessuna puntata	0	0	0	0	0	
20	18	Puntata	1	0	-1	-1	-1	1 : Street(4-6)
21	17	Puntata	1	0	-1	-1	-2	1 : Street(4-6)
22	17	Puntata	1	0	-1	-1	-3	1 : Street(4-6)
23	6	Puntata	1	12	-1	11	8	1 : Street(4-6)
24	32	Nessuna puntata	0	0	0	0	8	
25	13	Nessuna puntata	0	0	0	0	8	
26	20	Nessuna puntata	0	0	0	0	8	
27	30	Nessuna puntata	0	0	0	0	8	
28	6	Puntata	1	12	-1	11	19	1 : Street(4-6)
29	10	Nessuna puntata	0	0	0	0	19	

Il sistema verificherà automaticamente errori di sintassi, in più c'è la possibilità di fare il debug

Per controllare se il sistema si sta comportando esattamente come previsto potete utilizzare durante i test con permanenze ad hoc o reali lo storico puntate

Riepilogo sistema

Inizializza

Memorizza 18 terzine

Controlla l'esito delle puntate (Risultati)

Se non si sta giocando verifica se ci sono le condizioni per giocare (Verifica)

Se si sta giocando piazza le puntate (Punta)

Qui di seguito alleghiamo il listato completo che potete anche scaricare.

Questa prima guida speriamo vi aiuti a familiarizzare con il programma e cominciare ad elaborare i vostri sistemi. Vi ricordo che essendo un programma costruito con blocchi di comandi il trucco sta nello riutilizzare parti di sistemi già programmati. Ad esempio la suddivisione delle routine può essere abbastanza standard.

Come per tutti i lavori e lavoretti, prima di partire a spron battuto è meglio avere le idee chiare, scrivendo tutto sulla carta e lavorando ad un blocco alla volta, per evitare di perdersi nei meandri della programmazione.

Una buona regola è iniziare a lavorare sempre su una copia nuova del sistema così che se perdetevi la bussola avete sempre una versione di backup abbastanza aggiornata.

```

system "Terzine in calore"
{
  il mio primo sistema
}
method "main"
begin
  while Starting a new Session
  begin
    Call "Richieste"
    Set list [1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2] record "Progressione" data
    copy list [Street(1-3),Street(4-6),Street(7-9),Street(10-12),Street(13-15),
              Street(16-18),Street(19-21),Street(22-24),Street(25-27),Street(28-30),
              Street(31-33),Street(34-36)] nel record "Terzine" layout
    Set flag "Gioca" false
    Put 1 record "Progressione" data index
  end

  while on Each Spin
  begin
    track last street 18 nel record "18 terzine" layout
    if record "18 terzine" layout count <18
    begin
      exit
    end
    if flag "Gioca" true
    begin
      call "Risultati"
    end
    Call "Verifica"
    if flag "Gioca" true
    begin
      put 100% record "Progressione" data sul record "Da Puntare" layout
    end
  end
end

end

method "Verifica"
begin
  if flag "gioca" true
  begin
    return
  end
  // inizializzazioni variabili di verifica
  clear record "Massimo" data
  Clear record "Num. Terzine in calore" data
  Set list [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] record "Conta nelle 18" data
  Set flag "gioca" false

```

```

//Inizio controllo
put 1 record "Terzine" layout index
loop until record "Terzine" layout index > record "Terzine" layout count
begin
  put 100% record "Terzine" layout index nel record "Conta nelle 18" data index
  put 1 record "18 terzine" layout index
  loop until record "18 terzine" layout index > record "18 terzine" layout count
  begin
    if record "Terzine" layout = record "18 terzine" layout
    begin
      add 1 record "Conta nelle 18" data
    end
    if record "Conta nelle 18" data > record "Massimo" data
    begin
      put 100% record "Conta nelle 18" data nel record "Massimo" data
    end
    add 1 record "18 terzine" layout index
  end
  add 1 record "Terzine" layout index
end

put 1 record "Conta nelle 18" data index
loop until record "Conta nelle 18" data index > record "Conta nelle 18" data count
begin
  if record "Conta nelle 18" data = record "Massimo" data
  begin
    add 1 record "Num. Terzine in calore" data
    put 100% record "Conta nelle 18" data index nel record "Terzine" layout index
    copy record "Terzine" layout nel record "Da Puntare" layout
  end
  add 1 record "Conta nelle 18" data index
end
// verifica condizioni sistema
if record "Massimo" data <=4
and record "Num. Terzine in calore" data =1
begin
  Set flag "gioca" true
end
end

```

```

method "Risultati"
begin
  if Any Street Bet won each
  begin
    set flag "Gioca" False
    put 1 record "Progressione" data index
  end
  else
  begin
    add 1 record "Progressione" data index
    if record "Progressione" data index > record "Progressione" data count
    begin
      set flag "Gioca" False
      put 1 record "Progressione" data index
    end
  end
  //Controlla stopwin e stoploss
  if Bankroll >= record "SW" data
  or Bankroll <= record "SL" data
  begin
    Stop Session
  end
end

method "Richieste"
begin
  group
  begin
    input data "Fermarsi in vincita di (es. 10) " record "SW" Data
    input data "Fermarsi in perdita di (es. -10) " record "SL" Data
  end
end

```