Appunti su Excel

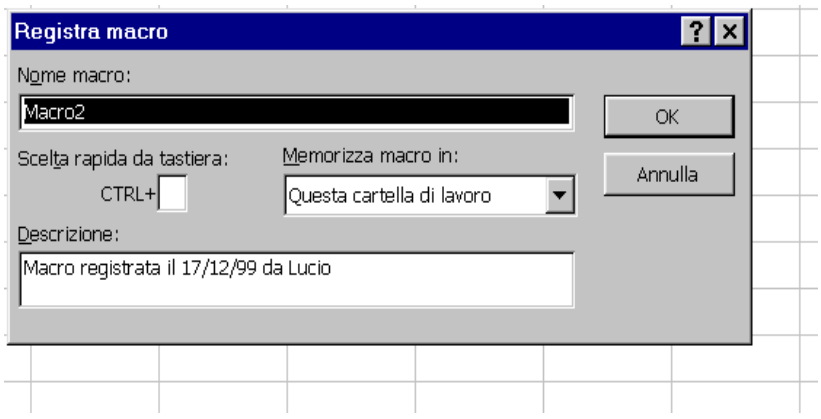
Introduzione:

Il corso si rivolge alle persone che già conoscono gli elementi di base del foglio elettronico e che intendono utilizzarlo per realizzare delle applicazioni, sfruttando anche l'uso delle macro e delle routines in visual basic.
Si è utilizzata una metodologia basata sugli esempi.
Alcuni esempi prendono spunto da articoli di riviste: SfondoArlecchino, Date, Tabella Pivot, Risolutore (auto). Purtroppo si tratta di riviste vecchie che ho da tempo cestinato e andando a memoria non vorrei citare la rivista sbagliata.
Gli altri sono sostanzialmente originali, talvolta frutto della necessità di risolvere reali problemi.
Nell'appendice ho riassunto alcune informazioni presenti nella guida in linea di Excel.
Autore: Lucio Borsato.

Sommario:

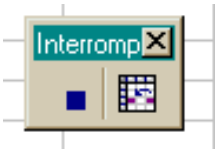
Registrazione di una macro:

Scegliere:
1. strumenti
2. macro
3. registra nuova macro

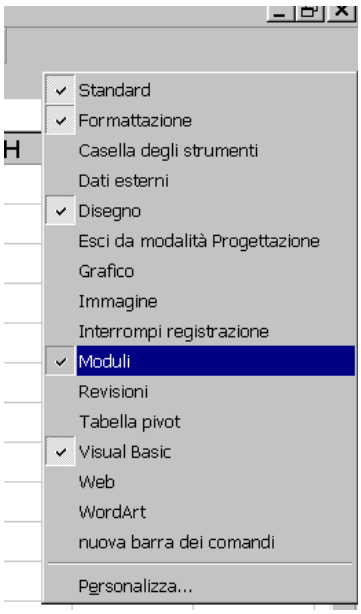


Digitare il nome **"Titoli"** come nome della macro.
Memorizzare la macro nella cartella **"personale"** per poterla utilizzare in qualunque altra cartella di lavoro.
La cartella "personale" è gestita da Excel in modo particolare.
All'apertura del programma viene comunque aperta, se esiste, e per non creare confusione all'utente rimane nascosta.
Viene salvata automaticamente in una direttrice definita al momento dell'installazione e non più modificabile: **C:\…….\XLSTART.**
Dal momento che sono sempre disponibili, per essere eseguite, tutte le macro appartenenti alle cartelle aperte e che la suddetta cartella è sempre aperta, memorizzare una macro in "personale.XLS" significa averla sempre a disposizione.

A questo punto compare la finestra che consentirà di interrompere la registrazione della macro.
Vogliamo ora registrare le seguenti operazioni:



- Selezioniamo il colore dello sfondo *giallo*.
- Selezioniamo il colore del carattere *blu*.
- Selezioniamo *grassetto* per il carattere.
- Selezionare la dimensione del carattere (12 pti)
- Interrompiamo la registrazione premendo il pulsante fine registrazione.

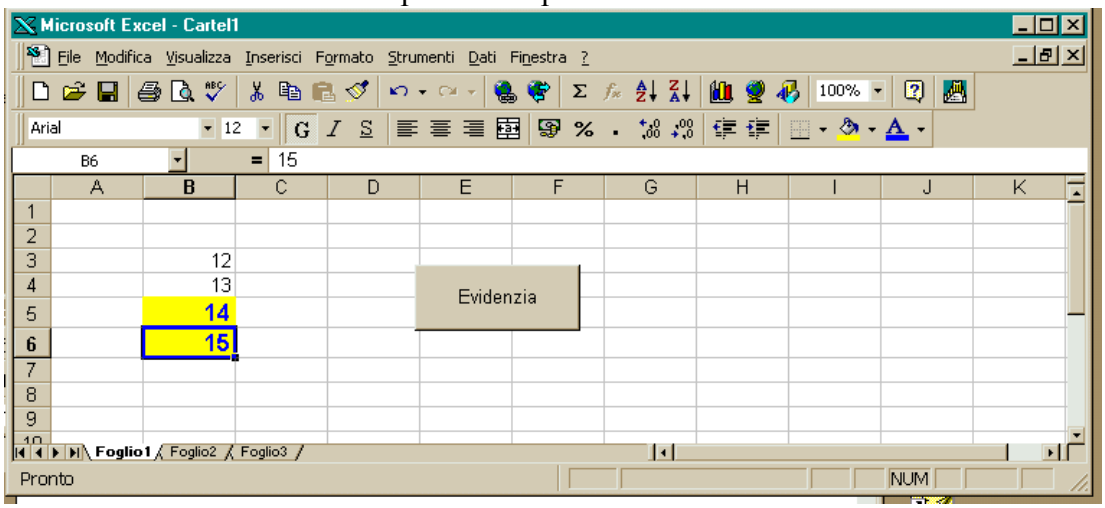


Per eseguire una macro, possiamo ad esempio associarla ad un pulsante di comando:
aprire la barra moduli (tasto destro sulla barra dei comandi, e poi spuntare la casella corrispondente a moduli).
Selezionare quindi pulsante e disegnarlo sul foglio di lavoro, nella posizione desiderata.
Associare quindi al pulsante la macro voluta "*Titoli*" e cambiare l'etichetta proposta (pulsante1) con una più significativa, ad esempio *Titoli*.

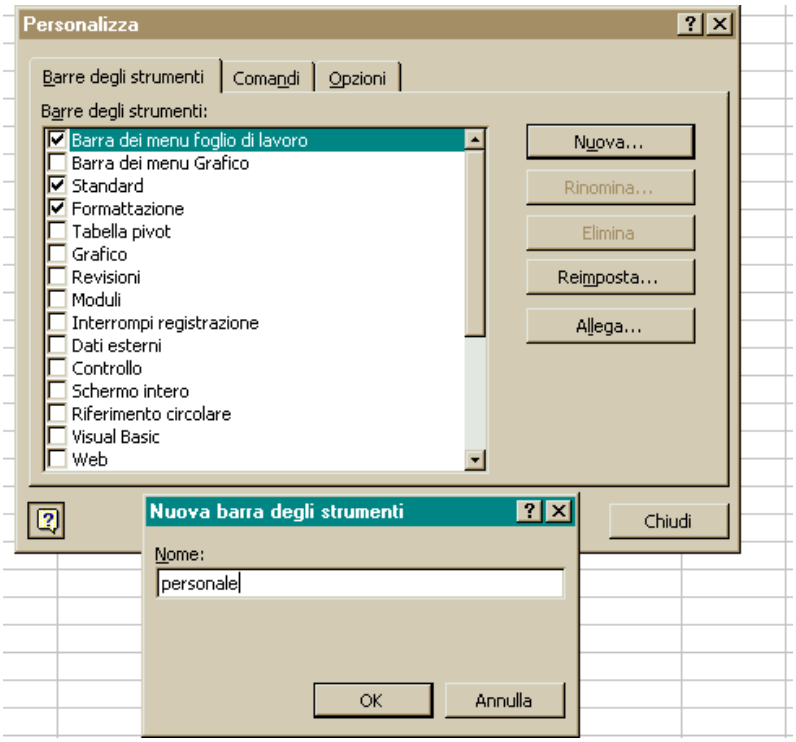
Per eseguire la macro:

- Selezioniamo alcune celle da evidenziare
- Premiamo il pulsante associato alla macro

Ed ecco il risultato, dopo aver evidenziato le celle b5 e b6 e aver premuto il pulsante evidenzia.

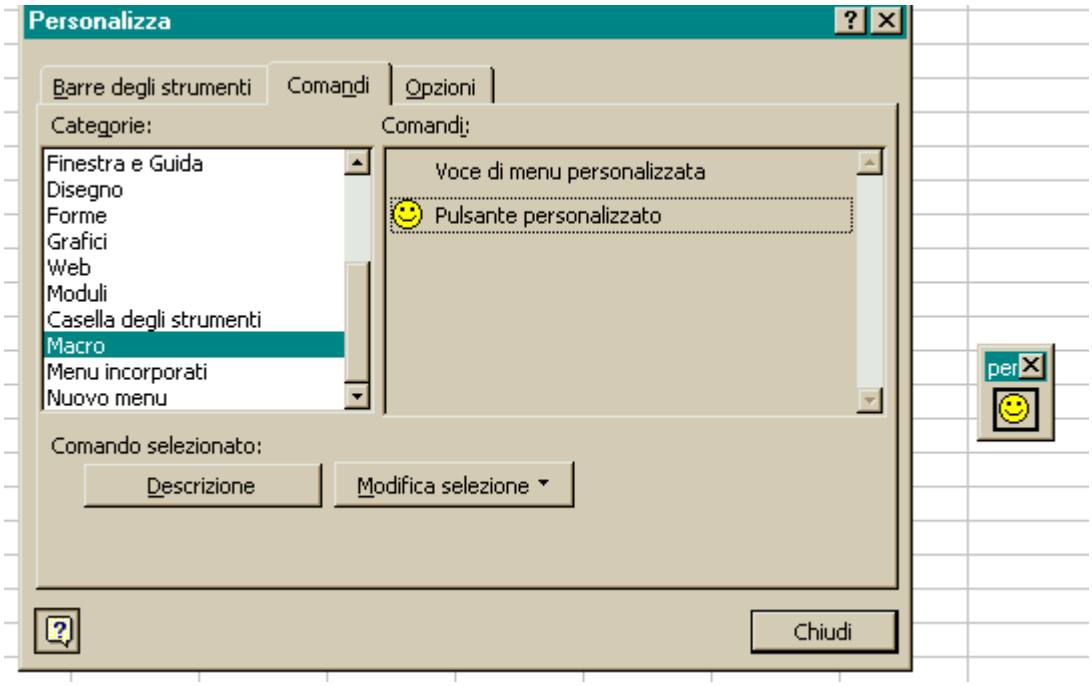


Se invece vogliamo avere a disposizione questa nuova macro in una barra degli strumenti, utilizzabile in tutte le cartelle, possiamo procedere così:

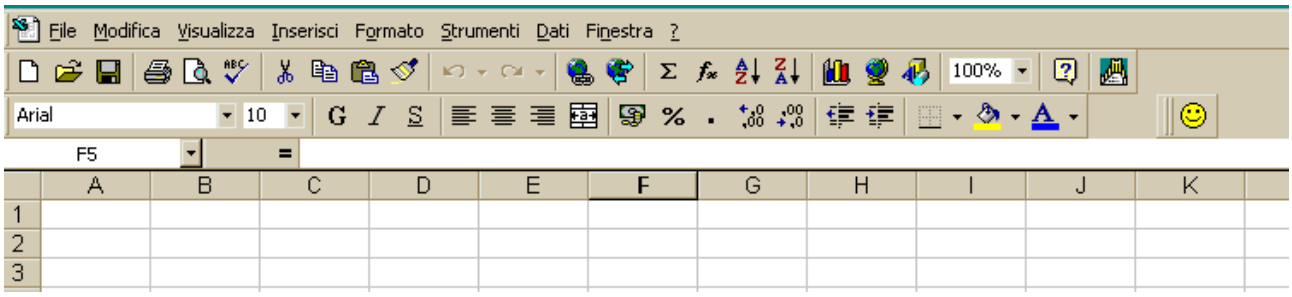


- *Visualizza\Barra degli strumenti\personalizza:*

Scegliamo *nuova* e poi digitiamo il nome personale,
OK,
selezioniamo la linguetta comandi,
evidenziamo *macro* e trasciniamo il pulsante personalizzato nella barra personale,*CHIUDI*,
Premiamo il nuovo pulsante (sole che ride)
Associamo il nome della macro Evidenzia, della cartella personale



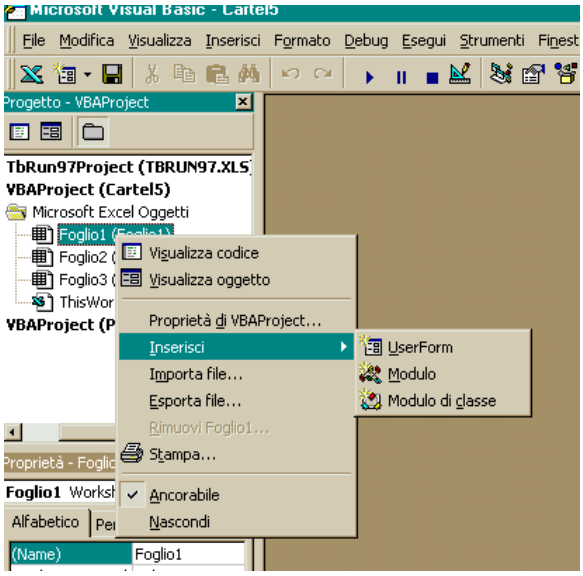
Trasciniamo la nuova barra degli strumenti dove preferiamo
Ora le barre degli strumenti si presenteranno cosi:



e la nuova macro sarà disponibile in tutte le cartelle.

Funzioni e Sub

Vogliamo ora provare a scrivere ed eseguire una Funzione ed una Sub.
Una routine **Sub** è una serie d'istruzioni racchiuse tra le istruzioni **Sub** e **End Sub** e che non restituisce un valore.
Una **Function** è una serie d'istruzioni racchiuse tra **Function** ed **End Function** e può restituire un valore. Scegliamo *Visualizza/Barra degli strumenti* e spuntiamo la casella *Visual Basic*
Entriamo in *editor di Visual Basic*
Tasto destro del mouse, *inserisci modulo*

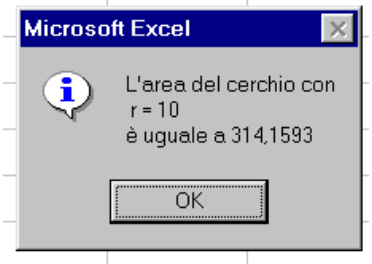


Possiamo ora nel modulo iniziare a scrivere la nostra funzione (**AreaCerchio**) e la Sub **Prova** che serve a provare tale funzione.

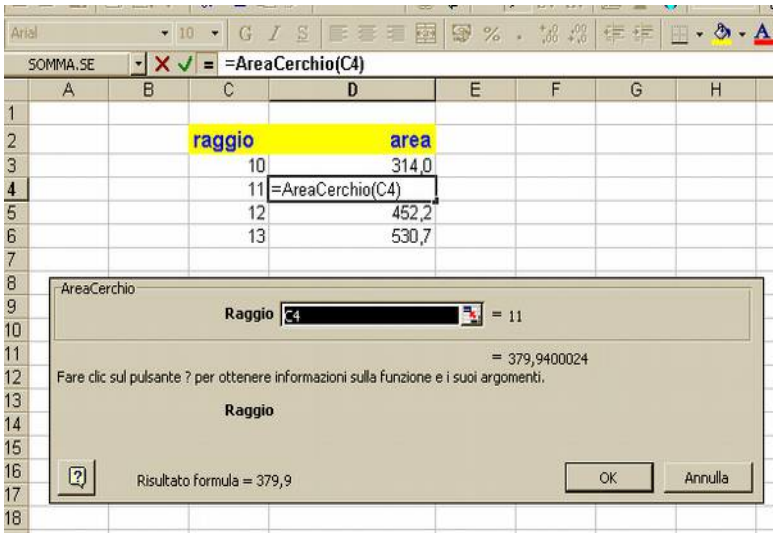
```
' sub utilizzata per provare la funzione AreaCerchio
Sub prova()
Dim Dato As Single
Dim Area As Single
'introduzione dati
Dato = InputBox(prompt:="Inserisci il raggio")
'calcolo l'area del cerchio utilizzando la funzione AreaCerchio e passandole il valore del raggio
Area = AreaCerchio(Dato)
'fornisce la risposta
risposta = "L'area del cerchio con" & Chr(13) & " r = " & Dato & _
   Chr(13) & "è uguale a " & Area
MsgBox prompt:=risposta, Buttons:=vbOKOnly + vbInformation
End Sub

================================================================

'calcola l'area del cerchio, dato il raggio
Function AreaCerchio(ByVal Raggio As Single) As Single
```

```
'il programma chiamante passerà una copia del valore del Raggio e non l'indirizzo, dato che Raggio è preceduto dall'opzione ByVal
Dim PiGreco As Single
'tre diversi modi per memorizzare la costante PiGreco:

'si attribuisce alla variabile PiGreco il valore 3,14
PiGreco = 3.14
'ricava la costante pigreco dalla funzione di Excel PI.GRECO()
'si noti che:
'   a) bisogna far precedere il nome della funzione dalla parola Application
'   b) si utilizza il nome della funzione della versione inglese
'      PI.GRECO() ===> PI()
PiGreco = Application.Pi()
'si calcola PiGreco utilizzando le funzioni del Basic
PiGreco = 4 * Atn(1)
'si calcola l'area del cerchio
AreaCerchio = PiGreco * Raggio ^ 2
End Function
```

Siamo ora in grado di provare la nostra Sub e la Function che verrà chiamata.
Strumenti/Macro/Macro
Selezioniamo il nome della macro **prova**
Esegui
Inseriamo il raggio nella finestra che compare
OK
In risposta otterremo il messaggio seguente.

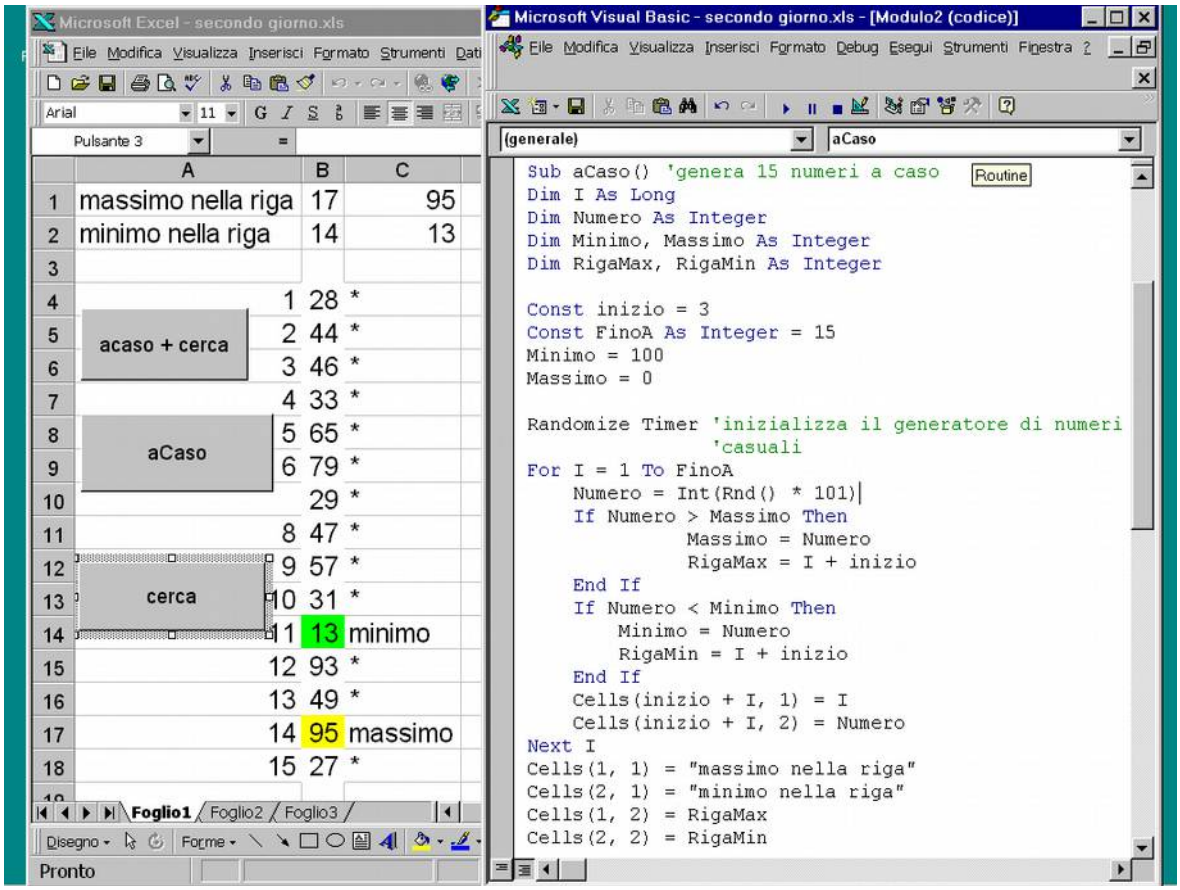


Possiamo anche utilizzare la funzione **AreaCerchio** nel foglio elettronico, in aggiunta a quelle fornite da Excel.
Ricordiamoci che se vogliamo utilizzare tale funzione in qualsiasi foglio di lavoro, dobbiamo memorizzarla in un modulo della cartella **person**.



Sub aCaso()

Nella prima *colonna*, a partire dalla quarta riga, registra un progressivo da 1 a 15.
Genera 15 numeri casuali compresi tra 0 e 100 e li inserisce nella seconda *colonna* del foglio di lavoro.
Nella prima e seconda *riga* del foglio di lavoro, inserisce rispettivamente il minimo ed il massimo dei numeri generati, nonché la riga corrispondente.
Evidenzia in verde e giallo il minimo ed il massimo
L'istruzione **Const Inizio = 3** permette di utilizzare la parola Inizio al posto del numero 3 in tutte le istruzioni della sub. Volendo cambiare il 3 in 5 in tutte le istruzioni in cui compare, sarà sufficiente modificare l'istruzione Const.



L'istruzione **Randomize Timer** inizializza il generatore di numeri casuali, utilizzando il timer del computer per stabilire l'elemento da cui partire.
Notiamo la presenza del gruppo **for I = 1 to FinoA ……. next I**
Tutte le istruzioni racchiuse all'interno vengono eseguite 15 volte, dal momento che FinoA vale 15.
Ogni volta che il ciclo viene eseguito, I viene incrementato di una unità.
L'istruzione **Int(Rnd()*101)** fornisce un numero intero compreso tra 0 e 101.
L'istruzione **Rnd()** fornisce un numero casuale non intero compreso tra zero ed 1.
Rnd()*101 permette di ottenere un numero casuale, con decimali maggiore o uguale a zero e minore di 101.
Int( ) restituisce la parte intera dell'argomento.
Le istruzioni **if Then Else Endif** servono per memorizzare il massimo ed il minimo dei numeri generati e le righe corrispondenti.
L'istruzione **Cells(Inizio + I, 2) = Numero**
consente di scrivere nella cella corrispondente alla
riga = Inizio + I
colonna = 2
del foglio di lavoro attivo, il numero casuale generato.

La sub Cerca() permette di individuare le celle contenenti i numeri massimi e minimi e li evidenzia, colorando lo sfondo di giallo e verde, e scrivendo nella cella a fianco rispettivamente "massimo" e "minimo".
Le istruzioni Racchiuse tra Do Until Cells(I + inizio, 1) *e* Loop *vengono eseguite un certo numero di volte, finchè la cella di coordinate I+Inizio ed 1 non risulta vuota.*
Dal momento che Inizio vale 3 e che ad I viene assegnato il valore 1 prima del gruppo Do, la prima volta viene esaminata la cella riga 4, colonna 1.
Le istruzioni If Then Else *permettono di memorizzare le righe corrispondenti al massimo ed al minimo numero letto.*
Non appena si incontra una cella vuota, si esce dal gruppo Do.
A questo punto (conoscendo le righe corrispondenti al massimo ed al minimo è possibile evidenziarli).
L'istruzione Cells(RigaMax, 2).Interior.Color = vbYellow permette di cambiare la proprietà del colore ***dello***

```
End Sub
Sub cerca()
'individua il massimo e il minimo dei quindici numeri
'e li evidenzia colorando lo sfondo delle celle
Dim I As Long

Dim Minimo, Massimo As Integer
Dim RigaMax, RigaMin As Integer

Const inizio = 3
I = 1
Minimo = 100
Massimo = 0
Do Until Cells(I + inizio, 1) = ""
    Cells(I + inizio, 2).Interior.Color = vbWhite
    Cells(I + inizio, 3) = "*"
    If Cells(I + inizio, 2) > Massimo Then
        Massimo = Cells(I + inizio, 2)
        RigaMax = I + inizio
    End If
    If Cells(I + inizio, 2) < Minimo Then
        Minimo = Cells(I + inizio, 2)
        RigaMin = I + inizio
    End If
I = I + 1
Loop    'fine gruppo istruzioni Do Until
Cells(RigaMax, 3) = "massimo"
Cells(RigaMin, 3) = "minimo"
Cells(RigaMax, 2).Interior.Color = vbYellow
Cells(RigaMin, 2).Interior.Color = vbGreen
End Sub
```

sfondo della cella di coordinate RigaMax, 2 assegnandole il colore giallo. vBYellow è una costante numerica corrispondente al colore giallo.
L'istruzione Cells(RigaMax, 3) = "massimo" permette di memorizzare la scritta "massimo" nella cella di coordinate RigaMax, 3.

Sub SfondoArlecchino()

```
'colora lo sfondo delle celle selezionate in modo casuale
Dim Verde, Rosso, Blu As Integer
Dim C As Range
' la funzione Rnd() genera un numero casuale compreso tra zero ed uno
'la funzione RGB permette di ottenere un colore mescolando i tre componenti fondamentali
'C in questo caso rappresenta un oggetto Range, cioè una cella
'l'istruzione For ……. next permette di ripetere un gruppo di istruzioni per ogni C appartenente alla zona
'selezionata
.Interior.Color è una proprietà dell'oggetto range (colore dello sfondo)
.Value permette di scrivere qualcosa nella cella selezionata (in questo caso il numero corrispondente al
colore dello sfondo)
For Each C In Selection
```

```
Rosso  =      Int(Rnd() * 256)
Verde  =      Int(Rnd() * 256)
Blu    =      Int(Rnd() * 256)
       C.Interior.Color = RGB(Rosso, Verde, Blu)
       C.Value = RGB(Rosso, Verde, Blu)
Next
End Sub
```

Ed ecco cosa si ottiene eseguendo la macro, su un gruppo di celle selezionate.

9734324	12995914	13681155	6884277	6277852
974838	8805875	9899460	10439799	4670373
9884628	3860988	4127409	16718728	9634989
13376025	4918088	15879521	4679418	10823977
11954536	3514963	1348911	4384629	4874441
10527211	9378157	14018993	15371013	8433006
5928579	935271	1047102	8215907	4290855
2656928	8497136	13114211	10010741	3474645
5511698	8978464	13863848	11350292	2513780
8908212	6734102	3505526	9836628	1699115

Function ConvNumLett(ByVal Numero As Currency) As String

Funzione che converte un numero in una stringa

Ad esempio 127 ➔ centoventisette

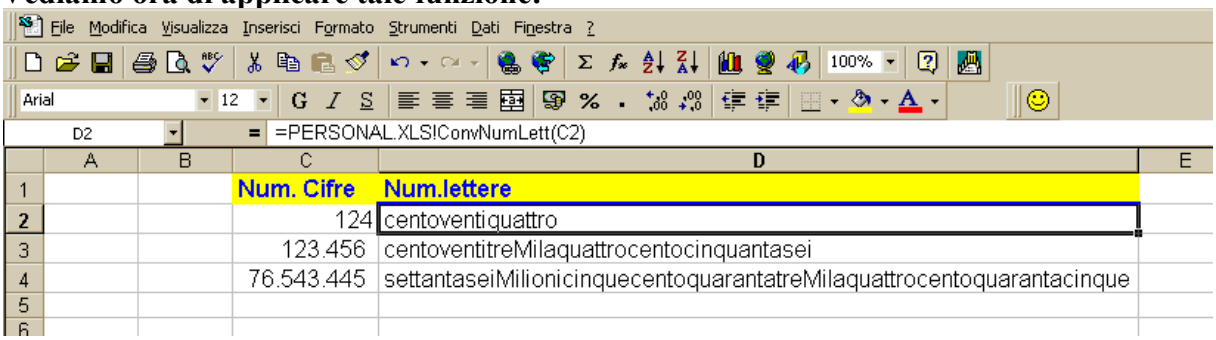
'Si riceve il valore di numero e non 'l'indirizzo; di conseguenza non si modifica il campo Numero del programma chiamante, l'alternativa è ByVal.

```
Dim Stringhetta      As String
Dim DesUnità(20)     As String      'vettore di 21 elementi di tipo stringa con indice compreso tra 0 e 20
Dim DesDecine(10)    As String
Dim DesEsp(4)        As String
Dim DesEspUno(4)     As String
Dim Esponente        As Integer
Dim decine           As Byte
Dim unità            As Byte
Dim centinaia        As Byte
Dim Resto            As Integer
Dim NumTre           As Integer
Dim OldNumero        As Currency
OldNumero = Numero
Stringhetta = ""
DesUnità(0) = ""
DesUnità(1) = "uno"
DesUnità(2) = "due"
DesUnità(3) = "tre"
DesUnità(4) = "quattro"
DesUnità(5) = "cinque"
DesUnità(6) = "sei"
DesUnità(7) = "sette"
DesUnità(8) = "otto"
DesUnità(9) = "nove"
```

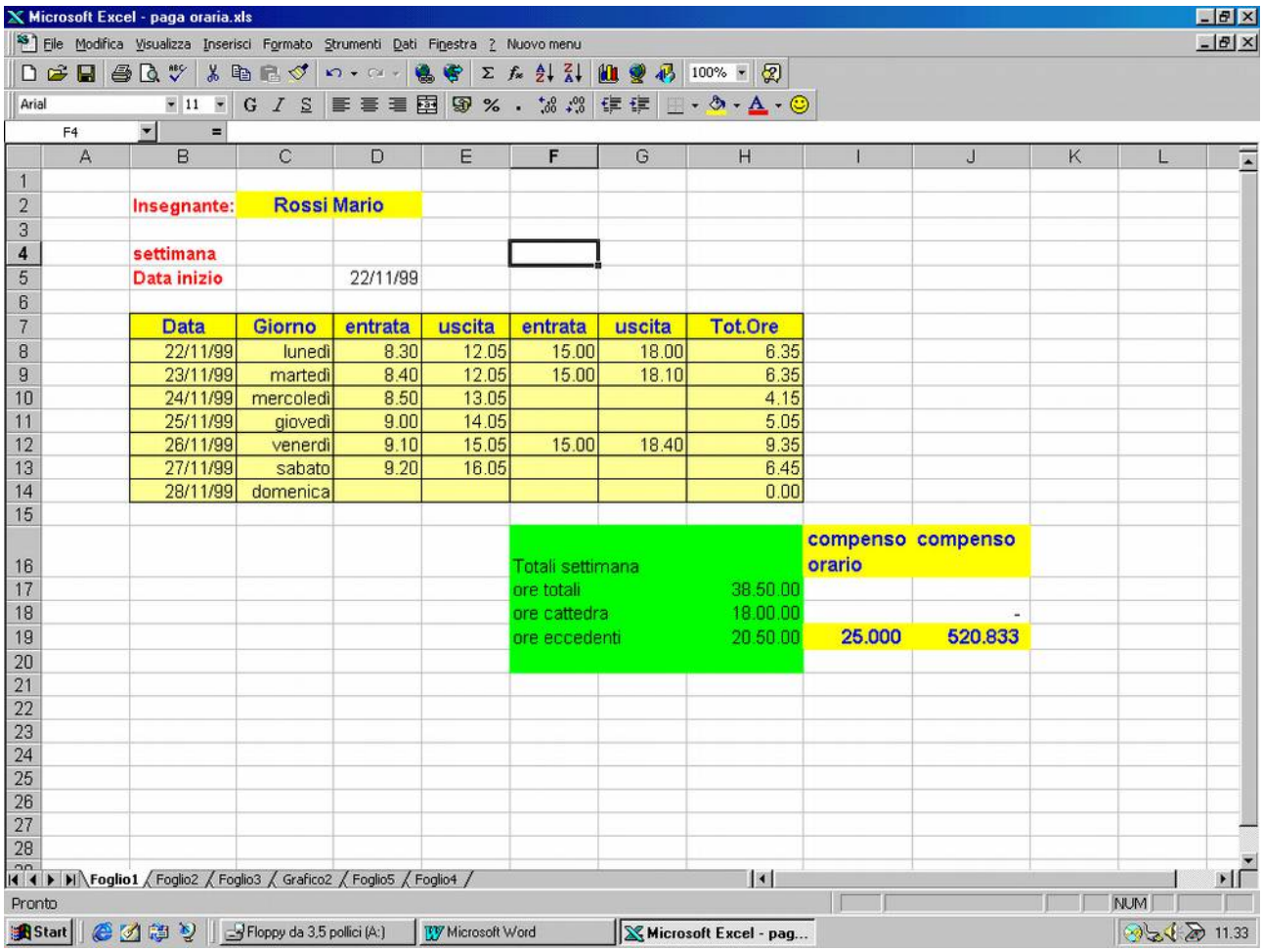
```
DesUnità(10) = "dieci"
DesUnità(11) = "undici"
DesUnità(12) = "dodici"
DesUnità(13) = "tredici"
DesUnità(14) = "quattordici"
DesUnità(15) = "quindici"
DesUnità(16) = "sedici"
DesUnità(17) = "diciassette"
DesUnità(18) = "diciotto"
DesUnità(19) = "diciannove"
DesDecine(2) = "venti"
DesDecine(3) = "trenta"
DesDecine(4) = "quaranta"
DesDecine(5) = "cinquanta"
DesDecine(6) = "sessanta"
DesDecine(7) = "settanta"
DesDecine(8) = "ottanta"
DesDecine(9) = "novanta"
DesDecine(10) = "cento"
DesEsp(0) = ""
DesEsp(1) = "Mila"
DesEsp(2) = "Milioni"
DesEsp(3) = "Milardi"
DesEspUno(0) = ""
DesEspUno(1) = "Mille"
DesEspUno(2) = "unMilione"
DesEspUno(3) = "unMiliardo"

'se il numero è negativo, aggiunge il prefisso "meno"
'e cambia il numero in positivo
If Numero < 0 Then
Numero = 0 - Numero
        stringhetta = "meno"
endif

'Tratta tre cifre alla volta, partendo da quelle più significative
For Esponente = 4 To 0 Step -1
'esponente = 4 ==> migliaia di miliardi
 'esponente = 3 ==> miliardi
 'esponente =2 ==> milioni
 'esponente = 1 ==> migliaia
 'esponente = 0 ==> unità
'NumTre contiene tre cifre partendo da migliaia di miliardi
   NumTre = Int(Numero / 1000 ^ Esponente)

Select Case NumTre
        Case Is < 0
                Stringhetta = "non converto numeri negativi"
        Case Is = 0
                Stringhetta = stringhetta & ""
        Case Is = 1000
                Stringhetta = "mille"
        Case Is < 1000
                 'converte le centinaia
                centinaia = Int(NumTre / 100)
                If centinaia = 0 Then
                Else
```

```
                    If centinaia > 1 Then
                        Stringhetta = Stringhetta & DesUnità(centinaia)
                    Else
                    End If
                    Stringhetta = Stringhetta & "cento"
                End If
                'converte le decine
                Resto = NumTre - centinaia * 100
                decine = Int(Resto / 10)
                If decine > 1 Then
                    Stringhetta = Stringhetta & DesDecine(decine)
                    'converte le unità
                    Resto = Resto - decine * 10
                    If Resto = 1 Or Resto = 8 Then
                        Stringhetta = Left(Stringhetta, Len(Stringhetta) - 1) _
                        & DesUnità(Resto)
                    Else
                        Stringhetta = Stringhetta & DesUnità(Resto)
                    End If

                Else
                    'converte un numtre inferiore a venti
                    Stringhetta = Stringhetta & DesUnità(Resto)
                End If
            Case Else
                Stringhetta = "non sono in grado di convertirlo"
        End Select

'aggiunge mille o mila alle migliaia
'milione o milioni
'miliardo o miliardi

If NumTre = 1 Then
                Stringhetta = Left(Stringhetta, Len(Stringhetta) - 3)
                Stringhetta = Stringhetta & DesEspUno(Esponente)
Else
        If NumTre > 1 Then
                Stringhetta = Stringhetta & DesEsp(Esponente)
        End If
End If
Numero = Numero - Int(Numero / 1000 ^ Esponente) * 1000 ^ Esponente
Next Esponente
If OldNumero = 0 Then Stringhetta = "zero"
ConvNumLett = Stringhetta
End Function
```

Vediamo ora di applicare tale funzione:



date



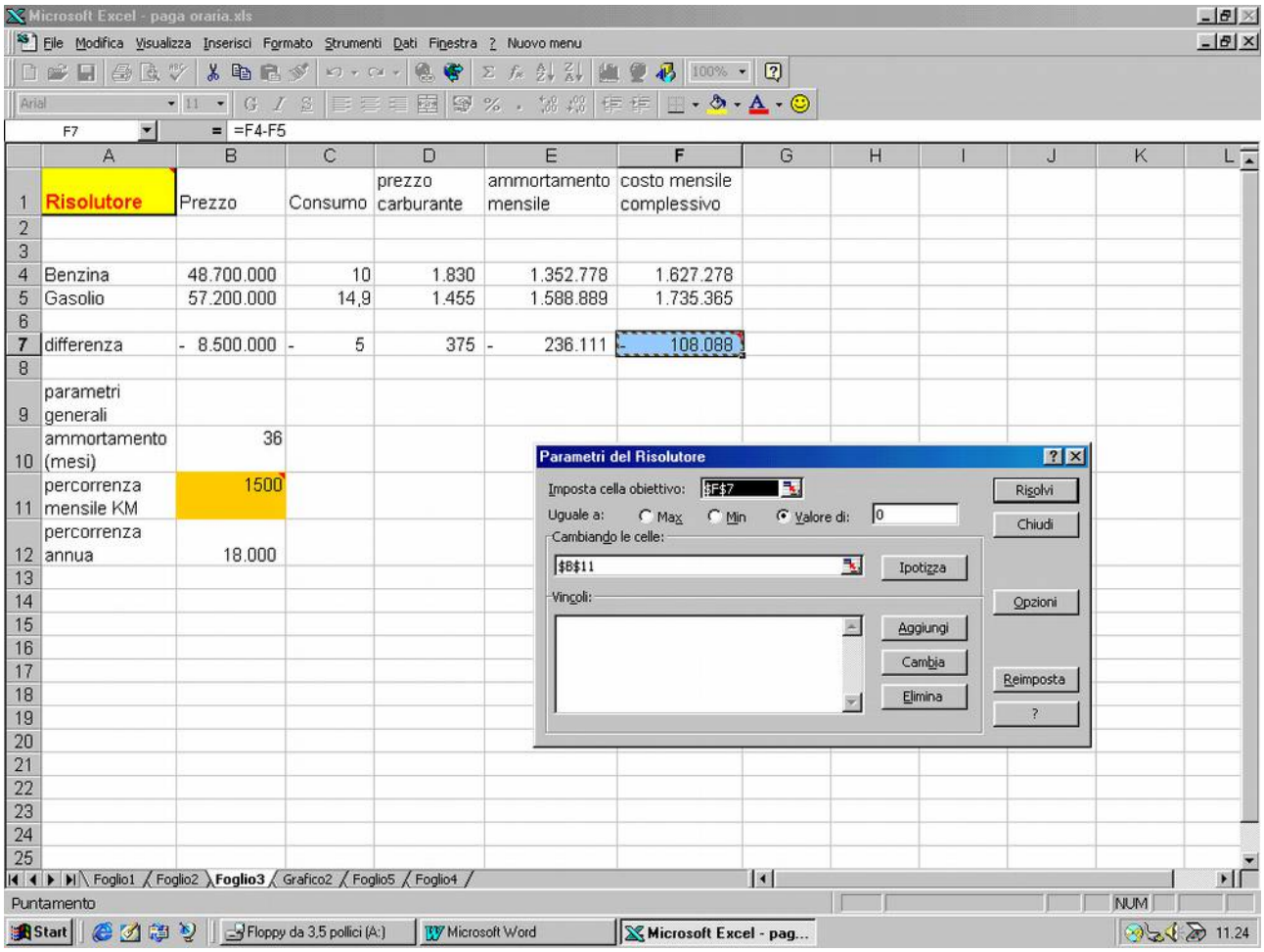
1. cella B8: =D5
2. cella B9: =B8+1
3. copiare la formula della cella B9, nelle celle B10…..B14
4. Cella C8: utilizzare la funzione **"GIORNO.SETTIMANA(B8,tipo)"**, che restituisce un numero da 1 a 7 corrispondente al giorno della settimana. Tipo permette di considerare come primo giorno il lunedì o la domenica.
5. Scegliere il formato personalizzato "**gggg**", per visualizzare lunedì, martedì, ecc. ecc…
6. Idem per le celle C9….C14
7. Per introdurre le ore utilizzare un formato ora, ad esempio 80:00
8. Tenere presente che introducendo un'ora o una data si memorizza un numero seriale, le ore zero del primo giorno del secolo corrispondono al numero 1.
9. L'unità di misura è il giorno.
10. Le ore **SEI** del primo giorno del secolo vengono registrate con il numero 1+ 6/24 = 1,25.
11. Se visualizziamo una cella contenente il numero 1,25, a seconda del formato prescelto potremo vedere:
    - 06:00 (con uno dei formati ora)
    - 01 gen 1990 (con uno dei formati data)
    - 1,25 con un formato numero
    - 01 gen 1990 06:00 con un diverso formato data/ora
12. Tenere presente che anche se leggiamo ore totali 38:50, in realtà il numero memorizzato è: 38/24+50/(24*60) e di questo si deve tener conto anche nel calcolare il compenso_settimanale =

ore eccedenti le 18 per compenso orario➔ il compenso orario deve essere moltiplicato per 24, dato che l'unità di misura è il giorno.

13. Per visualizzare un numero di ore superiore a 24, utilizzare un formato personalizzato del tipo [h]:mm

14. Nel calcolare le ore lavorate nell'arco della giornata, tenere presente il caso in cui l'uscita avviene dopo le ore 24, utilizzare quindi la funzione **"SE".**

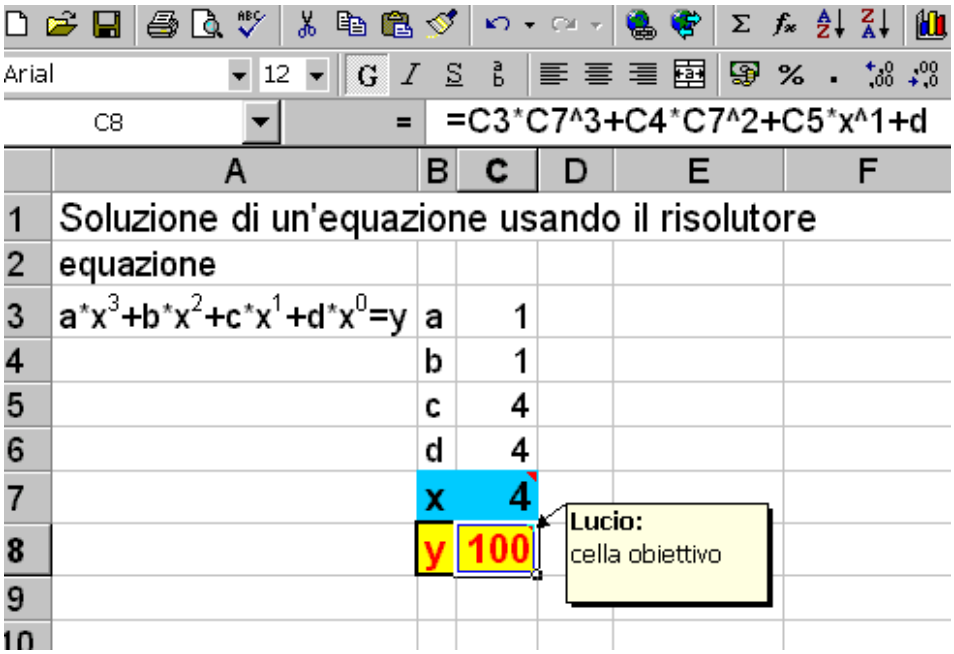
Utilizzo del risolutore:



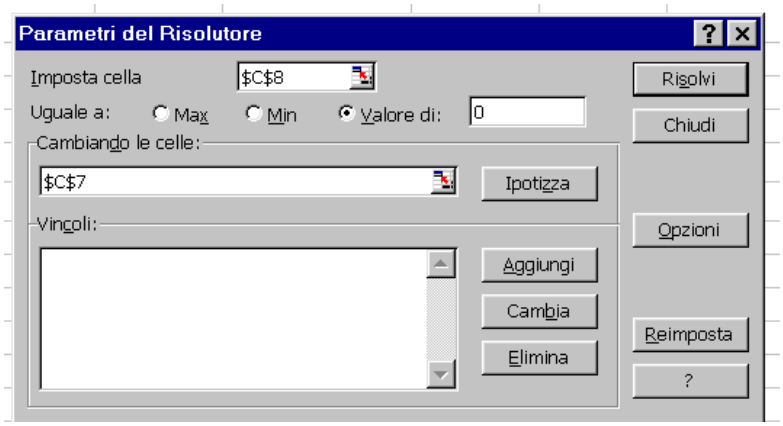
- La cella obiettivo è la F7
- Verrà calcolato automaticamente il valore della cella B11 (percorrenza mensile), che consente di realizzare l'obiettivo, cella F7=0.

Altro esempio di utilizzo del risolutore:

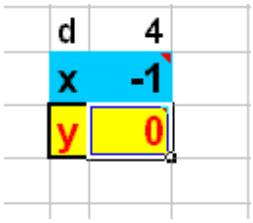
Soluzione di un'equazione, nel campo dei numeri reali.



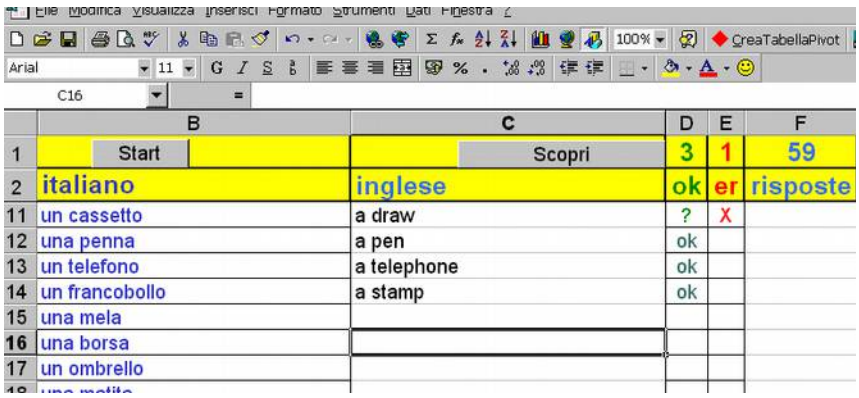
La cella C8 contiene la formula visualizzata nella barra delle formule e rappresenta la cella obiettivo, la cella C7 rappresenta invece la grandezza che viene fatta variare.



Scegliere strumenti, risolutore: Impostare la cella C8 Uguale al valore di zero, cambiando la cella C7.
Ed ecco il risultato che si ottiene:



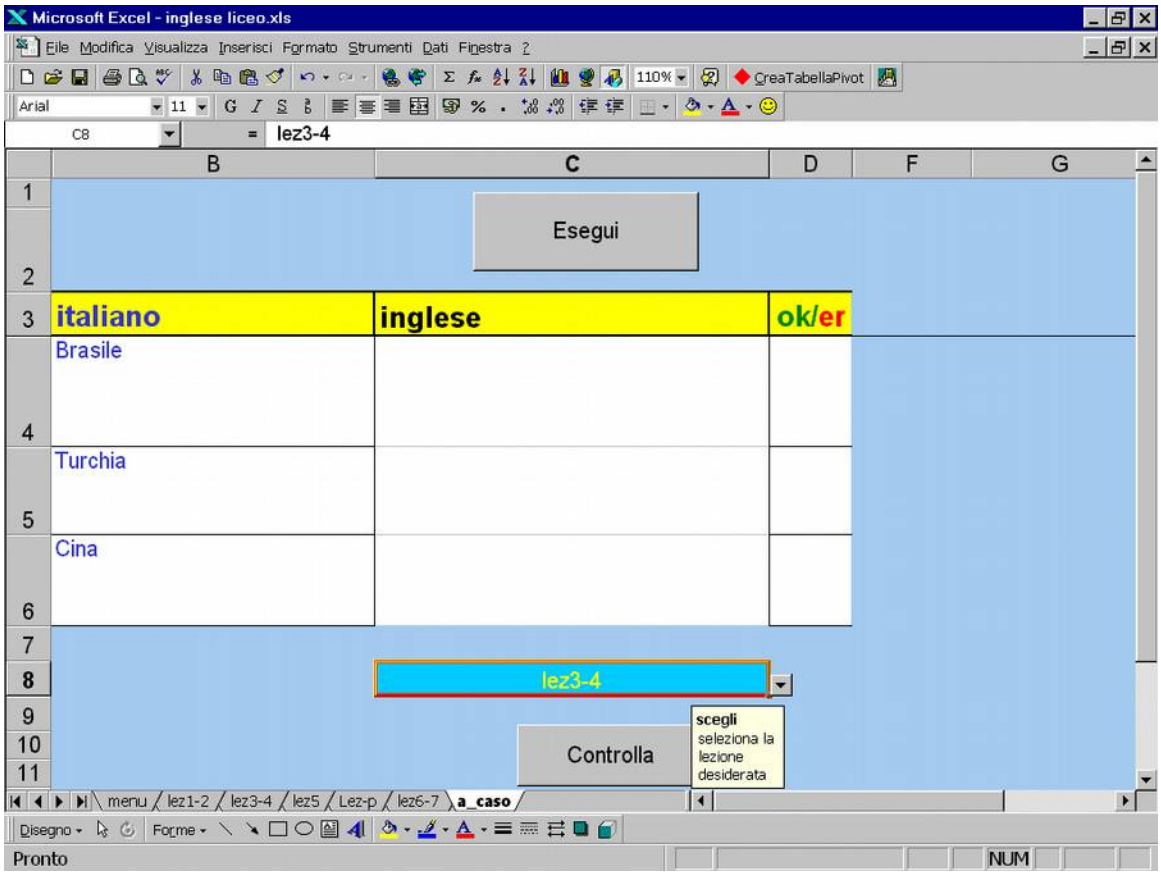
Quesiti a risposta chiusa:



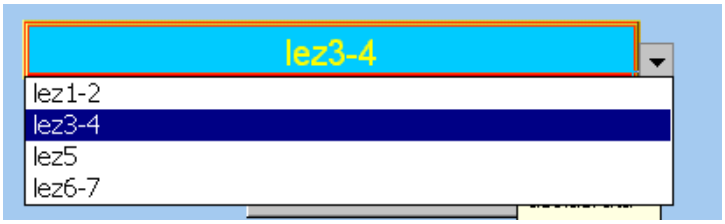
La colonna A, attualmente nascosta, contiene le risposte esatte; la colonna B le domande; la colonna C le risposte che darà l'allievo. Nelle colonne D ed E si segnalano le risposte esatte e quelle errate.
Al pulsante Start è associata una macro che cancella le risposte già date e permette di rieseguire l'esercizio dall'inizio. Al pulsante Scopri è associata una macro che permette di scoprire la colonna nascosta A e quindi di controllare quali avrebbero dovuto essere le risposte esatte.
Ed ecco la formula associata alle celle della colonna D:

La funzione Identico è sensibile a maiuscolo/minuscolo, a differenza del confronto con =

Vediamo ora come scegliere casualmente i quesiti, avendo a disposizione alcuni archivi (fogli).

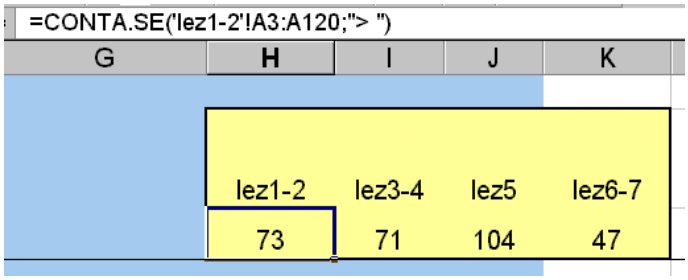


Vengono proposti tre quesiti ogni volta che si preme il tasto esegui.



Nella cella C8 si può selezionare l'archivio (il foglio) contenente le domande di un argomento, capitolo ecc. ecc. è sufficiente aprire la tendina e selezionare il foglio desiderato.

Per ottenere questo risultato occorre, in un'area del foglio, indicare i dati validi per la cella C8, e nella cella C8 scegliere dati/convalida e selezionare quindi l'area del foglio che contiene i dati validi (in questo caso le celle (H3-K3). La formula evidenziata nella cella H4 individua il numero di quesiti presenti nel foglio lez1-2, contando le celle dalla A3 alla 120 il cui contenuto è diverso da niente. (Si prevedono al massimo 117 quesiti per foglio).



Al pulsante Esegui è associata una SUB che seleziona tre quesiti a caso dal foglio di lavoro desiderato e li copia nel foglio attivo; naturalmente copia anche le risposte esatte, che rimangono nascoste nella colonna A.

```
Sub Esegui()
'   sceglie alcune domande a caso da uno degli archivi
' nuovo Macro
' Macro registrata il 30/10/99 da Lucio
'
    Sheets("a_caso").Select
   'Randomize
    Columns("a").Select
    Selection.EntireColumn.Hidden = True 'nasconde la colonna A
    ' se si desidera utilizzare il foglio lez1-2 come archivio
    'Cells(3,8) contiene il numero di quesiti di lez1-2
    Randomize Timer
    If Cells(8, 3) = "lez1-2" Then
        numero = Int((Cells(3, 8) - 3) * Rnd())
         numero = numero + 1
        'numero sarà compreso tra 1 ed il numero corrispondente al
        ' terzultimo quesito
        Sheets("lez1-2").Select ' si seleziona il foglio lez1-2
        ' si selezionano le celle corrispondenti alle colonne A, B C
        ' ed alle tre righe che vanno da numero a numero + 2
        Range(Cells(numero, 1), Cells(numero + 2, 3)).Select
        'si copiano negli appunti
        Selection.Copy
        Sheets("a_caso").Select 'si seleziona il foglio a_caso
        Range("A4").Select  'la cella A4
        ActiveSheet.Paste   'si incollano le celle zelezionate
        Set mioIntervallo = Worksheets("a_caso").Range("A4:C6")
    Else
    End If
    If Cells(8, 3) = "lez3-4" Then
```

Nelle celle D4, D5, D6 troviamo le solite formule che segnalano se la risposta è esatta o errata.
Per la formattazione delle tre celle si è utilizzato: formato/condizionale:

Tabella Pivot:

Microsoft Excel - Cartel1

File Modifica Visualizza Inserisci Formato Strumenti Dati Finestra ?

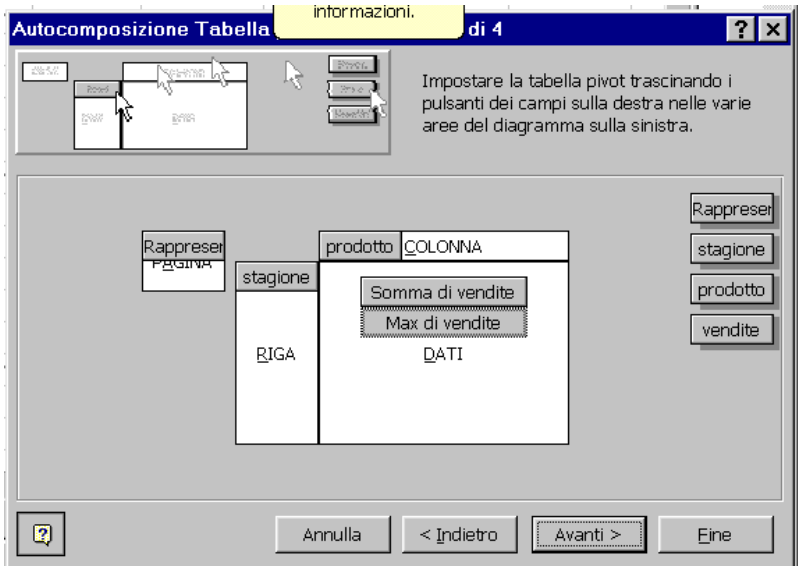
Arial 11 G I S

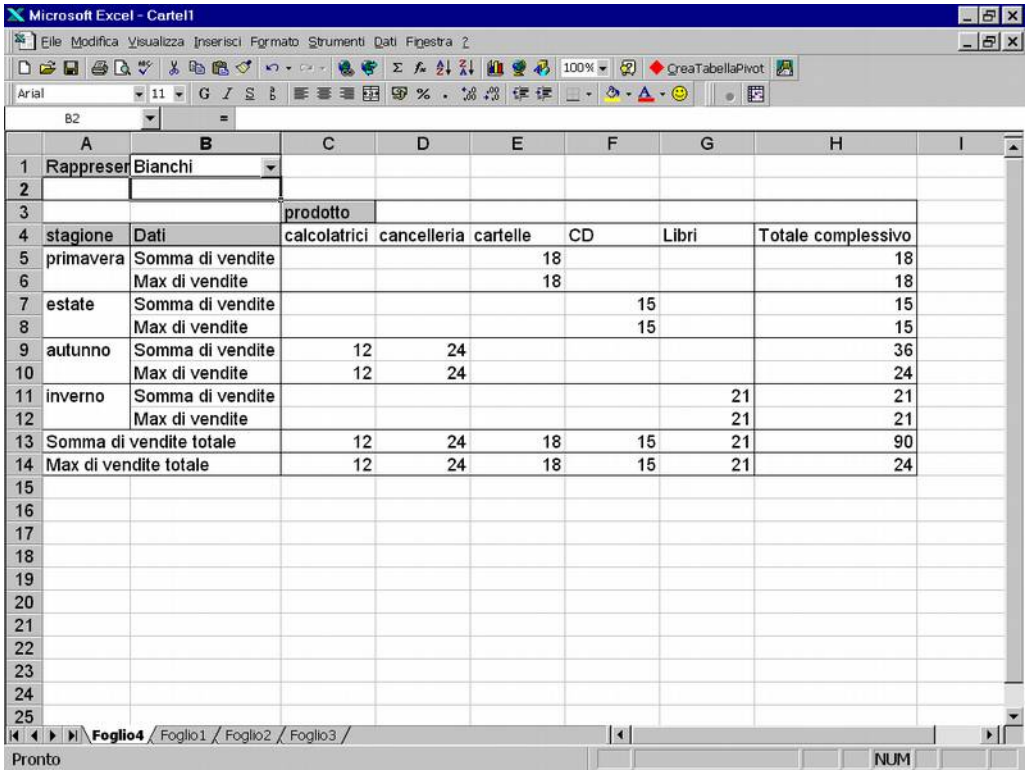
G8 =

	A	B	C	D	E
1	Rappresentante	stagione	prodotto	vendite	
2	Bianchi	autunno	calcolatrici	12	
3	Bianchi	autunno	cancelleria	24	
4	Bianchi	primavera	cartelle	18	
5	Bianchi	estate	CD	15	
6	Bianchi	inverno	Libri	21	
7	Rossi	primavera	calcolatrici	22	
8	Rossi	estate	cancelleria	19	
9	Rossi	inverno	cartelle	13	
10	Rossi	primavera	CD	10	
11	Rossi	inverno	CD	25	
12	Rossi	autunno	Libri	16	
13	Verdi	inverno	calcolatrici	17	
14	Verdi	primavera	cancelleria	14	
15	Verdi	estate	cartelle	23	
16	Verdi	autunno	CD	20	
17	Verdi	estate	Libri	11	
18	Verdi	primavera	Libri	26	
19					
20					

Supponiamo di voler costruire una tabella Pivot, per evitare di dover ripetere l'operazione ogni volta che i dati vengono modificati, registriamo le operazioni in una macro. Rendere attiva una cella contenente i dati, quindi selezionare *dati/Report tabella Pivot:* scegliere *Elenco o databased Microsoft Excel*, accettare l'intervallo di celle selezionato automaticamente, se corretto, quindi indicare come elaborare i dati.
Scegliamo Rappresentante in pagina, stagione in riga, prodotto in colonna e vendite in dati (due volte)
Verrà proposta la somma delle vendite, accettiamo la prima e modifichiamo la seconda in max dopo un doppio click nel campo somma vendite2.

Quindi procediamo con *avanti* e poi con *fine*, indicando dove deve essere incollata la tabella Pivot.
Ed ecco il risultato:
è possibile selezionare un rappresentante o tutti, agendo sulla tendina posta in alto.





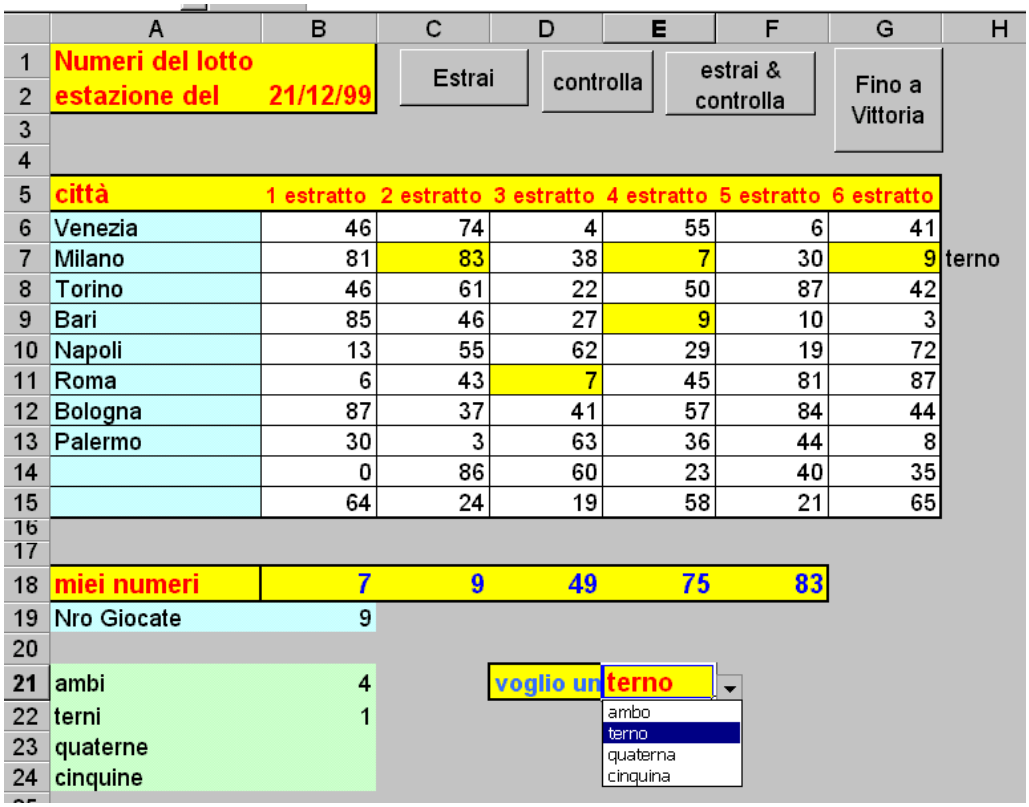
	A	B	C	D	E	F	G	H	I
1	Rappreser	Bianchi							
2									
3			prodotto						
4	stagione	Dati	calcolatrici	cancelleria	cartelle	CD	Libri	Totale complessivo	
5	primavera	Somma di vendite			18			18	
6		Max di vendite			18			18	
7	estate	Somma di vendite				15		15	
8		Max di vendite				15		15	
9	autunno	Somma di vendite	12	24				36	
10		Max di vendite	12	24				24	
11	inverno	Somma di vendite					21	21	
12		Max di vendite					21	21	
13	Somma di vendite totale		12	24	18	15	21	90	
14	Max di vendite totale		12	24	18	15	21	24	
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									

Ed ecco la macro registrata:

```
Sub Pivot()
'
' Pivot Macro
' Macro registrata il 19/12/99 da Lucio
'

'
    Range("B7").Select
    ActiveSheet.PivotTableWizard SourceType:=xlDatabase, SourceData:= _
        "Foglio1!R1C1:R18C4", TableDestination:="R3C6:R5C6", TableName:= _
        "Tabella_pivot2"
    ActiveSheet.PivotTables("Tabella_pivot2").AddFields RowFields:=Array( _
        "stagione", "Dati"), ColumnFields:="prodotto", PageFields:="Rappresentante"
    With ActiveSheet.PivotTables("Tabella_pivot2").PivotFields("vendite")
        .Orientation = xlDataField
        .Position = 1
    End With
    With ActiveSheet.PivotTables("Tabella_pivot2").PivotFields("vendite")
        .Orientation = xlDataField
        .Name = "Max di vendite"
        .Function = xlMax
    End With
End Sub
```

è possibile rieseguirla in qualunque momento, dopo aver modificato i dati della tabella, senza dover rieseguire tutte le operazioni, ma semplicemente con *macro/esegui/Pivot*

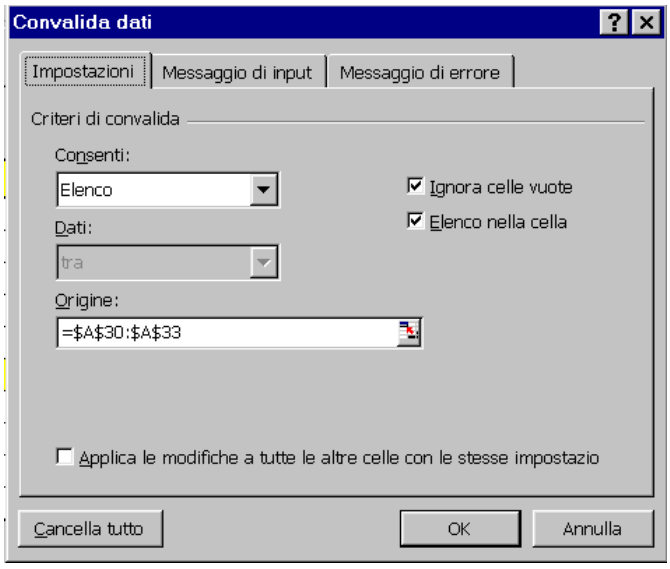
Il Gioco del Lotto



	A	B	C	D	E	F	G	H
1	Numeri del lotto		Estrai	controlla	estrai & controlla		Fino a Vittoria	
2	estazione del	21/12/99						
3								
4								
5	città	1 estratto	2 estratto	3 estratto	4 estratto	5 estratto	6 estratto	
6	Venezia	46	74	4	55	6	41	
7	Milano	81	83	38	7	30	9	terno
8	Torino	46	61	22	50	87	42	
9	Bari	85	46	27	9	10	3	
10	Napoli	13	55	62	29	19	72	
11	Roma	6	43	7	45	81	87	
12	Bologna	87	37	41	57	84	44	
13	Palermo	30	3	63	36	44	8	
14		0	86	60	23	40	35	
15		64	24	19	58	21	65	
16								
17								
18	miei numeri	7	9	49	75	83		
19	Nro Giocate	9						
20								
21	ambi	4		voglio un	terno			
22	terni	1						
23	quaterne							
24	cinquine							

ambo
terno
quaterna
cinquina

Proviamo a simulare il gioco del lotto.
La sub Estrai, simula l'estrazione su tutte le ruote. La sub controlla segnala quali numeri giocati sono stati estratti e li segnala colorando di giallo lo sfondo delle celle corrispondenti. Inoltre, in caso di vincite, le segnala con una scritta in corrispondenza alla ruota fortunata. Il pulsante Estrai e controlla manda in esecuzione una sub che esegue in successione le due precedenti.
Infine è possibile rieseguire l'estrazione ed il controllo più volte fino al raggiungimento di una certa vincita indicata nella cella E21 segnalando il numero delle estrazioni (cella B19) e le vincite complessive (celle da B21 a B24). Tale sub ha il nome di finoaVittoria.
Per evitare errori di digitazione nella cella E21, è stata utilizzata l'0pzione *convalida dati* del menù *dati*.



Si è scelto *Elenco* per la casella consenti (criteri di convalida) e le celle da A30 ad A33 per l'origine dei dati.

29	
30	ambo
31	terno
32	quaterna
33	cinquina
34	

Ed ecco il contenuto delle celle in questione.

Vediamo ora la sub Estrai():

```
Sub estrai()
'
' estrai Macro
' Macro registrata il 20/12/99 da Lucio estrae i numeri del lotto
'
Dim Riga, Colonna As Integer
Dim I As Integer
Dim Numero As Integer
Dim Diverso As Boolean
Dim Ruota(6) As Integer
Const Primo As Integer = 1

For Riga = 6 To 15
    Randomize Timer

    For I = 1 To 6
        Diverso = True
        Do
            Diverso = True
            Numero = Int(Rnd() * 91)
            For J = 1 To I - 1
                If Numero = Ruota(J) Then Diverso = False

            Next J
        Loop Until Diverso
        Colonna = Primo + I
        Ruota(I) = Numero
        Cells(Riga, Colonna) = Numero
    Next I
Next Riga
End Sub
```

Il gruppo *For Riga 6 to 15* viene eseguito 10 volte (estrazione dei numeri corrispondenti alle dieci città). Per ognuna di queste righe devono essere estratti 6 numeri compresi tra 0 e 90, senza ripetizione. Il gruppo *Do .... Loop Until Diverso* permette di ripetere l'strazione finchè il numero non è diverso dai precedenti. L'istruzione *Int(Rnd()*91)* permette di ottenere un numero intero, casuale, compreso tra 0 e 90. Tale numero viene confrontato con quelli già estratti nella stessa ruota (città) e memorizzati nel vettore *Ruota(J)*.
Il numero estratto, se valido, viene scritto nella colonna opportuna *(primo + I)*; il primo estratto finirà nella colonna 2.

```
Sub controlla() 'controlla se i numeri giocati sono usciti
Dim Indovinato, I, J, Riga, Colonna As Integer
For Riga = 6 To 15
    Indovinato = 0
    For I = 1 To 5
        Numero = Cells(18, I + 1)
           For Colonna = 2 To 7
                If Numero = Cells(Riga, Colonna) Then
                    Cells(Riga, Colonna).Interior.Color = vbYellow
                    Indovinato = Indovinato + 1
                Else
                    If I = 1 Then
                        Cells(Riga, Colonna).Interior.Color = vbWhite
                    Else
                    End If
                End If
            Next Colonna
    Next I
    Select Case Indovinato
    Case Is = 2
        Cells(Riga, 8) = "ambo"
        Cells(21, 2) = Cells(21, 2) + 1
    Case Is = 3
        Cells(Riga, 8) = "terno"
        Cells(22, 2) = Cells(22, 2) + 1
    Case Is = 4
        Cells(Riga, 8) = "quaterna"
    Case Is = 5
        Cells(Riga, 8) = "cinquina"
    Case Else
        Cells(Riga, 8) = ""
    End Select
```

La sub Controlla esamina una ruota alla volta (for riga 6 ..to 15) considera uno alla volta i numeri giocati (For I = 1 to 5) e li confronta con quelli estratti (colonne dalla 2 all 7), se corrispondono colora lo sfondo della cella di giallo (cells(Riga, Colonna).Interior.Color=vbYellow)) e incrementa il contatore Indovinato. Dopo aver esaminato tutti i numeri giocati, se per quella ruota sono almeno due, viene segnalata la vincita, scrivendo la sigla corrispondente (ambo o terno o…) in corrispondenza della ruota ed aggiornando il riepilogo delle vincite (si incrementa di 1 la cella opportuna).
Se per quella ruota non si sono realizzate vincite si cancella una eventuale scritta relativa ad una precedente estrazione.
il tutto verrà ripetuto per le ruote successive, quindi prima della fine Sub ci sarà l'istruzione *Next Riga.*

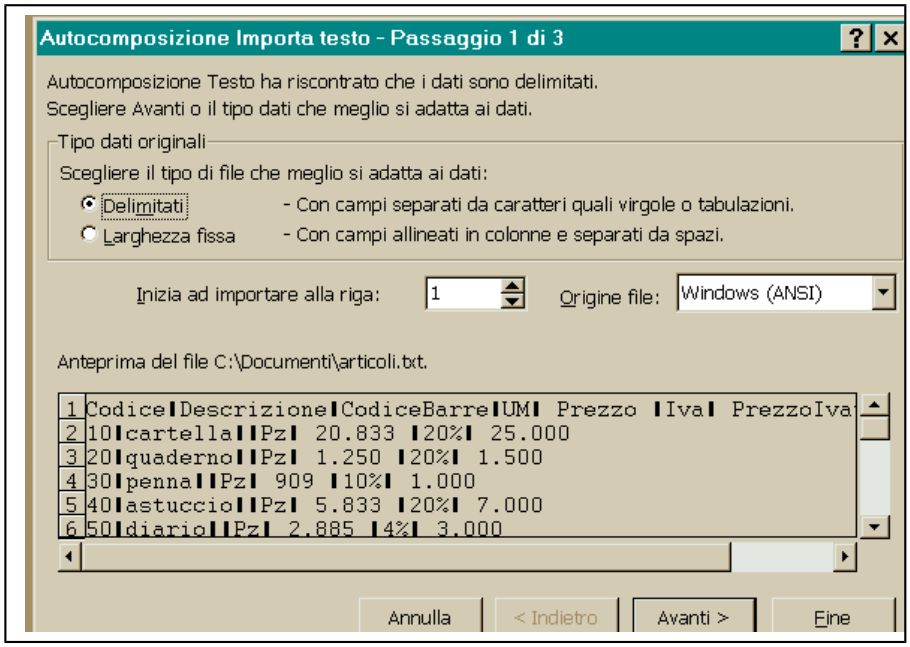
Vediamo ora le due Sub rimanenti:

```
Sub EstraiControlla()
estrai
controlla
End Sub
Sub FinoaVittoria()
For J = 19 To 25
    Cells(J, 2) = ""
Next J
If Cells(21, 5) = "ambo" Then Riga = 21
If Cells(21, 5) = "terno" Then Riga = 22
If Cells(21, 5) = "quaterna" Then Riga = 23
If Cells(21, 5) = "cinquina" Then Riga = 24
Do Until Cells(Riga, 2) > 0
    EstraiControlla
Loop
End Sub
```

EstraiControlla esegue in successione la procedura *estrai* e poi *controlla*. Come si può notare, la sua scrittura è estremamente semplice e compatta.
FinoaVittoria riesegue le estrazioni ed i controlli fino al raggiungimento di un cero obiettivo programmato (vincita di una certa categoria), digitato nella cella riga21, colonna5.
Il primo gruppo For J = 19 To 25 cancella le vincite precedenti prima di procedere alle estrazioni. Le quattro istruzioni IF permettono di stabile in quale riga (categoria) dovrà comparire una vincita per potersi considerare soddisfatti ed interrompere le estrazioni.
Il gruppo Do Until ….. Loop riesegue estrazioni e controlli fino a che nella cella(Riga, 2) non si è ottenuta una vincita.

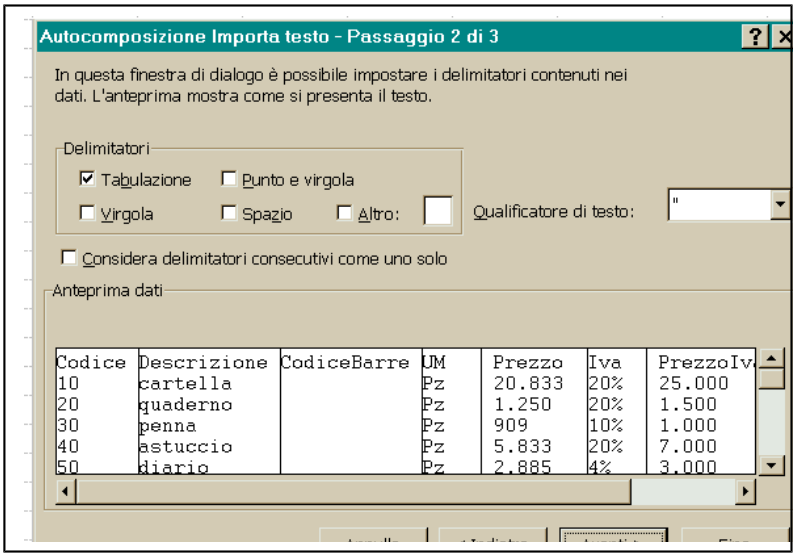
Fatturazione:

Come prima operazione prevediamo di importare i dati riguardanti gli articoli, prelevandoli da un archivio articoli.txt.
Selezioniamo File/Apri e poi il nome del file di testo. Scegliamo di considerare i dati separati da tabulazioni, dal momento che così sono stati memorizzati.
L'alternativa principale è costituita da dati a lunghezza fissa.



Successivamente possiamo vedere un'anteprima di quello che sarà il risultato dell'importazione. Possiamo anche modificare alcuni parametri.
Ed ecco il risultato dell'importazione:

Codice	Descrizione	CodiceBarre	UM	Prezzo	Iva	Prezzolvato
10	cartella		Pz	20.833	20%	25.000
20	quaderno		Pz	1.250	20%	1.500
30	penna		Pz	909	10%	1.000
40	astuccio		Pz	5.833	20%	7.000
50	diario		Pz	2.885	4%	3.000
60	compasso		Pz	25.000	20%	30.000
70	album		Pz	7.273	10%	8.000
80	gomma		Pz	1.818	10%	2.000
90	temperino		Pz	1.545	10%	1.700
100	mine		Pz	818	10%	900
110	zainetto		Pz	41.667	20%	50.000



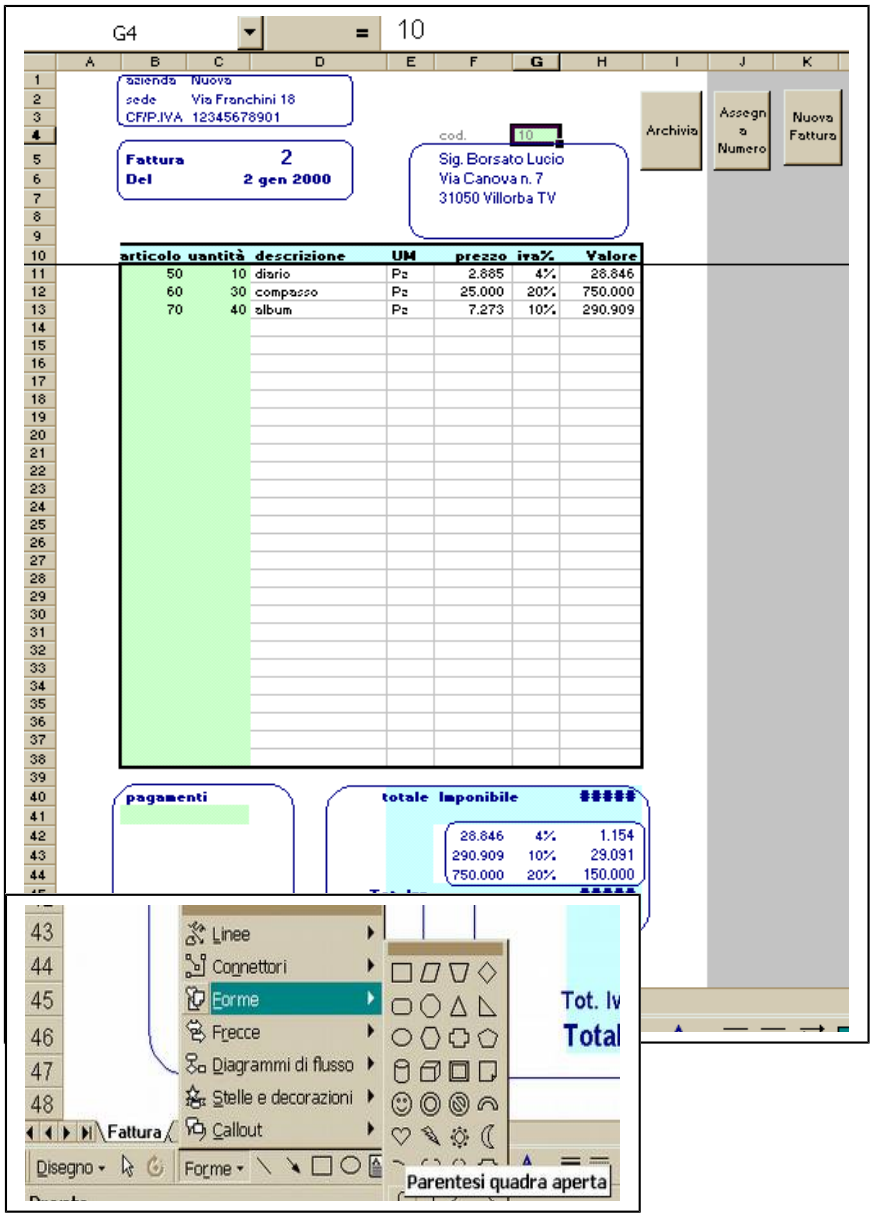
Un secondo archivio necessario è quello con i dati dei clienti. Utilizzeremo quindi un foglio di lavoro con il nome clienti.

	B	C	D	E	F	G	H	I	J	K
1	**RagioneSociale**	**Via**	**N**	**Cap**	**Comune**	**Provincia**	**Tel**	**Fax**	**Cfiscale**	**P.Iva**
2	Borsato Lucio	Canova	7	31050	Villorba	TV	0422918659		BRSLCU53B14H523Q	
3	Rossi Mario	Piave	8	31100	Treviso	TV				
4	Bianchi Luciano	Aproino	9	31100	Treviso	TV				
5	Verdi Andrea	Baracca	10	31100	Treviso	TV				
6	Dante Alighieri	Trieste	11	31100	Treviso	TV				
7										

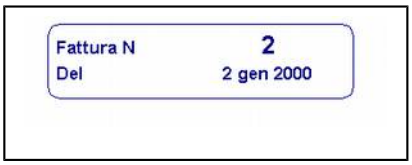
Vediamo ora il foglio di calcolo dove verrà elaborata la fattura.
In una zona libera del foglio, scriviamo alcuni dati che serviranno per automatizzare alcune operazioni e precisamente:

- L'ultimo numero utilizzato per le fatture
- Le aliquote I.V.A. che utilizzeremo.
- Le descrizioni delle condizioni di pagamento che potremo utilizzare.

	M	N	O	P	Q
1		UltimoNro	iva%	iva%	iva%
2		2	10%	20%	4%
3					
4		pagamenti			
5		Rimessa diretta			
6		Contanti			
7		R.B. fine mese			
8					
9					



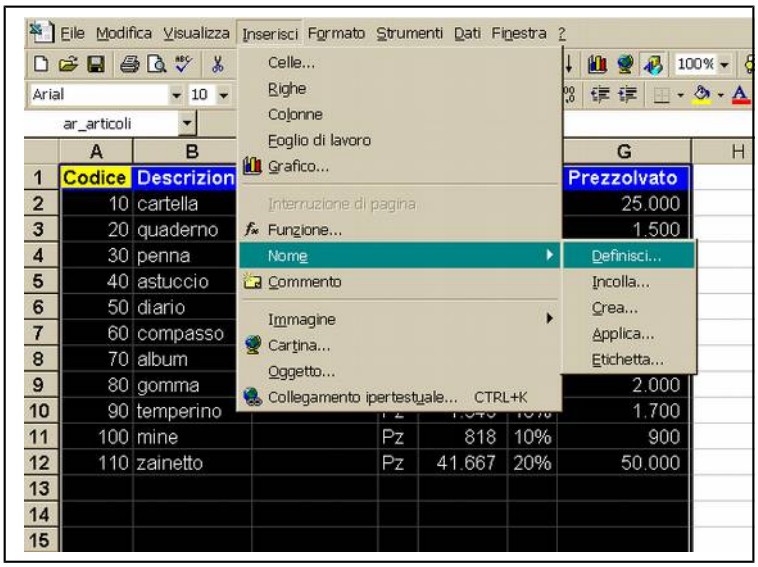
Le celle con sfondo verde sono quelle che l'utente deve riempire, tutto il resto sarà calcolato automaticamente. Il primo riquadro in alto a sinistra contiene i dati della ditta che fattura, dati che normalmente non vengono cambiati.
Il secondo riquadro a sinistra contiene il numero e la data della fattura. Il numero può essere assegnato automaticamente, mediante una semplice procedura che esamineremo successivamente e che viene



attivata premendo il pulsante *Assegna Numero*. La data è quella del sistema, infatti la casella corrispondente contiene la funzione: *=OGGI()* che restituisce la data del giorno.
La realizzazione dei riquadri richiede l'utilizzo della barra di strumenti per il disegno. Per realizzarlo sono stati utilizzati quattro elementi:

- una parentesi quadra aperta,
- una chiusa e
- due linee orizzontali.

Il tutto è poi stato raggruppato in modo da formare un solo oggetto. Gli altri riquadri sono stati ottenuti con una semplice operazione di copia/incolla e successivo ridimensionamento e posizionamento con il mouse.
Il riquadro in alto a destra corrisponde ai dati del cliente. È sufficiente digitare il codice del cliente. Ed ecco come si è ottenuto questo risultato:
Come prima operazione, per semplificare e rendere più leggibili le formule, abbiamo deciso di definire la parte del foglio di lavoro contenente gli articoli con il nome *ar_articoli*.
Per far questo selezioniamo *Inserisci/Nome/Definisci. Digitiamo* quindi il nome "*ar.articoli*" ed indichiamo a quale intervallo di celle si riferisce.



Allo stesso modo definiamo l'area che corrisponde all'archivio clienti utilizzando il nome *"ar.cliente"*

Vediamo come ottenere automaticamente il cognome e nome preceduto da Sig. Potremmo scrivere la seguente formula: *="Sig." & CERCA.VERT($G$4;ar_clienti;2;FALSO)*
Con la funzione CERCA.VERT si cerca il record della tabella *ar_clienti* avente lo stesso codice digitato nella cella *(G4)*, restituendo come risultato il campo della colonna 2, cioè la ragione sociale. Il quarto parametro False richiede una corrispondenza esatta per il codice che si cerca.
Il simbolo *&* permette di concatenare la stringa *Sig.* con la ragione sociale.
La formula inserita è leggermente più complessa, per gestire il caso in cui il codice digitato è errato oppure se non si digita nessun codice. In tal caso si preferisce lasciare vuota la cella, piuttosto che veder comparire una scritta che segnala un errore. Si è pertanto fatto ricorso ad altre due funzioni:

1. *VAL.ERRORE(...)* che restituisce *VERO* nel caso l'espressione tra parentesi dia luogo ad un errore (non si trova un record con il codice corrispondente)
2. *SE(condizione;se vero;se falso)*
   - La condizione è: si è verificato un errore utilizzando la funzione CERCA.VERTICALE
   - Se vero (si è verificato un errore) si aggiunge a *Sig.* la stringa vuota ""
   - Se falso si ottiene la ragione sociale con la funzione CERCA.VERTICALE

La formula finale è la seguente:
="Sig." & SE(VAL.ERRORE(CERCA.VERT($G$4;ar_clienti;2;FALSO));"";CERCA.VERT($G$4;ar_clienti;2;FALSO))
In modo analogo si costruiscono le formule per l'indirizzo del cliente.

```
Sub FatturaNum()
'assegna automaticamente un numero alla fattura
Dim Numero As Long
If Cells(5, 3) > 0 Then Exit Sub
Numero = Cells(2, 14) + 1

Cells(2, 14) = Numero
Cells(5, 3) = Numero

End Sub
```

La seguente procedura serve per assegnare automaticamente il numero della fattura.
La cella di coordinate 2, 14 contiene l'ultimo numero utilizzato.
La cella di coordinate 5, 3 è quella che corrisponde al numero della fattura in questione. Se tale numero è già stato

attribuito (automaticamente o manualmente) la funzione non esegue nessuna operazione (exit sub); in caso contrario l'ultimo numero viene incrementato di una unità e assegnato alla fattura.
Passiamo ora al dettaglio fattura.

articolo	quantità	descrizione	UM	prezzo	iva%	Valore
50	10	diario	Pz	2.885	4%	28.846
60	30	compasso	Pz	25.000	20%	750.000
70	40	album	Pz	7.273	10%	290.909

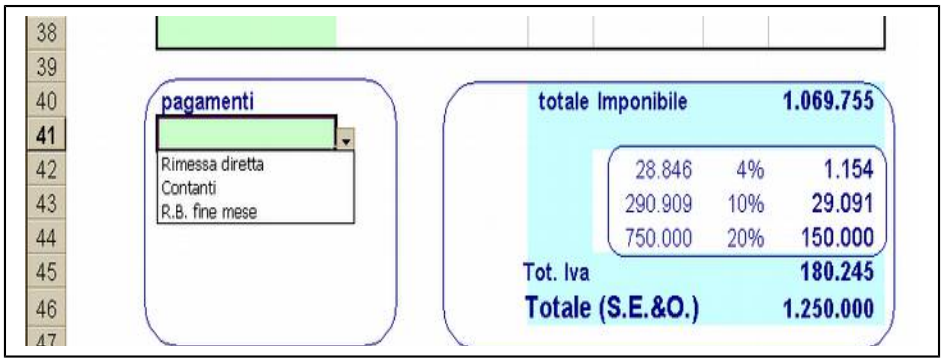
La sottile riga che divide le righe 10 ed 11 permette di visualizzare sempre la testata della fattura, dividendo la finestra in senso orizzontale in due parti. Per ottenere ciò le operazioni da effettuare sono:

- *click* sulla riga 11
- *Finestra/Blocca Riquadri*

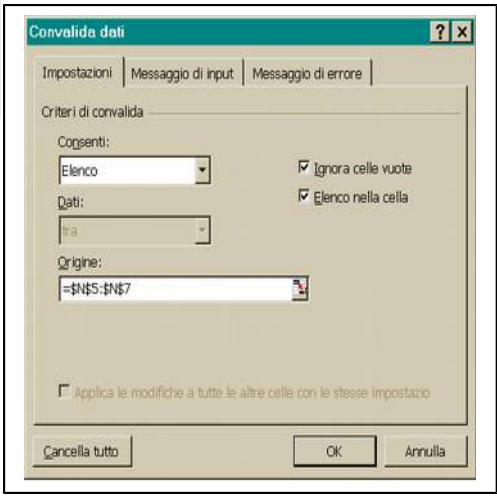
Il codice articolo deve essere uno di quelli appartenenti all'archivio articoli.
Vediamo la formula della cella di coordinate D11 dove compare la scritta diario.
Non gestendo gli errori la formula sarebbe la seguente:
=CERCA.VERT($B11;ar_articoli;2;FALSO)

- Il codice da cercare è quello della cella *B11*, l'indirizzo della colonna *B* è assoluto *$B*, per permettere una più agevole duplicazione della formula nei campi successivi.
- L'archivio in cui effettuare la ricerca è quello definito con il nome *ar_articoli*
- Il campo che si preleverà sarà quello della colonna *2* (descrizione degli articoli)
- La corrispondenza dei codici deve essere esatta (*FALSE*)

Volendo gestire anche gli errori, la formula assumerà la forma seguente:
=SE(VAL.ERRORE(CERCA.VERT($B11;ar_articoli;2;FALSO));"";CERCA.VERT($B11;ar_articoli;2;FALSO))
Analoghe sono le formule per inserire automaticamente UM, prezzo ed iva%.
Il campo Valore contiene la formula quantità * Prezzo; cioè:
*=C11*F11*

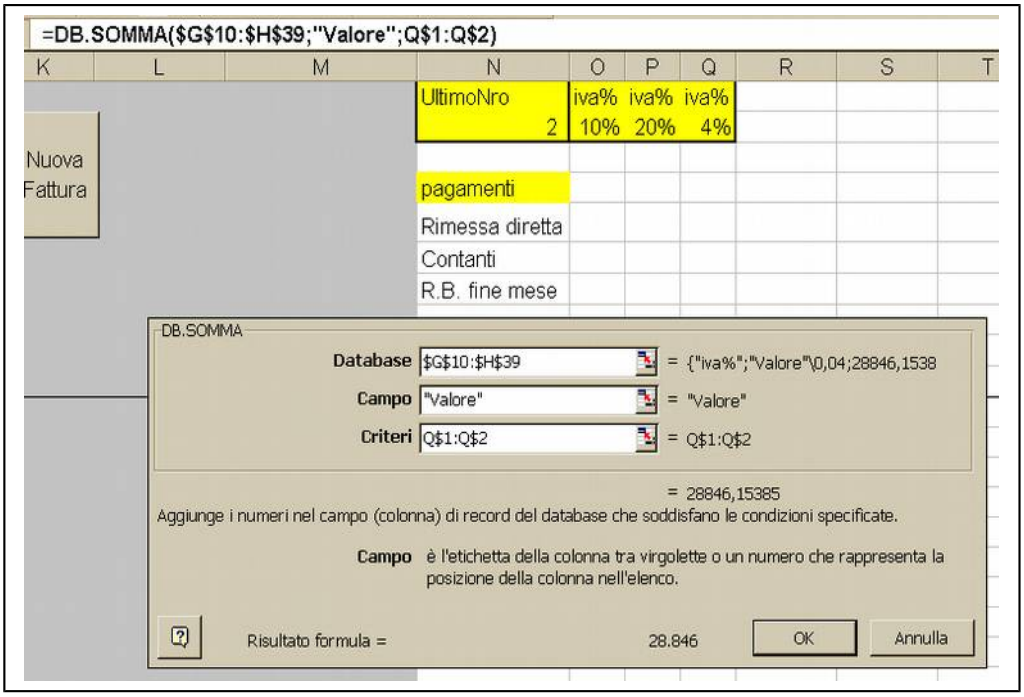


Analizziamo ora il piede della fattura:
La selezione del pagamento, mediante casella a discesa è stata ottenuta tramite:



Dati/Convalida: e successivamente scegliamo *impostazioni, Elenco* nella casella *Consenti*, e nella casella *Origine* indichiamo le celle N5:N7 che contengono le descrizioni delle modalità di pagamento.
La cella contenente il totale imponibile ha la semplice formula somma dei valori delle righe di dettaglio.

Troviamo quindi una tabellina con il dettaglio dell'I.V.A. (4, 10 e 20 percento); il totale dell'I.V.A. (Formula somma) ed il totale generale (imponibile + IVA).
Vediamo ora la formula che permette di ottenere l'imponibile con IVA al 4%.



=DB.SOMMA($G$10:$H$39;"Valore";Q$1:Q$2)
La funzione utilizzata è: *DB.SOMMA(Database;Campo;Criteri)*

- Il Database è costituito dalle righe di dettaglio, campi *IVA% e Valore*
- In campo digitiamo tra virgolette "*Valore*" indicando quale dato deve essere sommato
- In criteri indichiamo le celle *Q1:Q2*, in questo caso gli indirizzi delle righe 1 e 2 sono assoluti per permettere una agevole duplicazione della formula per le altre aliquote.

```
Sub NuovaFattura()
'
' NuovaFattura Macro
' Macro registrata il 02/01/00 da Lucio
' Cancella i dati della vecchia fattura
' e predispone per una nuova

'
    Range("C5:D5").Select
    Selection.ClearContents
    Range("G4").Select
    Selection.ClearContents
    Range("B11:C38").Select
    Selection.ClearContents
    Range("B41:C41").Select
    Selection.ClearContents
    Cells(11, 2).Select
End Sub
```

Si noti che l'etichetta in Q$1 (*iva%)* deve corrispondere all'etichetta della testata delle righe di dettaglio; stesso discorso per "Valore". Visualizziamo ora la macro che viene eseguita premendo il pulsante Nuova Fattura:
Si tratta semplicemente di cancellare i campi digitati dall'utente predisponendoli per una nuova fattura.
Vediamo ora di analizzare la *sub ArchiviaFattura(). Tale procedura permette di archiviare i dati della fattura appena compilata in un archivio (foglio di lavoro) che individuiamo con il nome (arch_fat). Per ogni fattura verranno archiviate n righe, corrispondenti alle righe di dettaglio e con l'aggiunta di alcuni dati della testata e del piede. La prima parte della sub memorizza i dati della fattura in alcuni campi o matrici.*
Nella seconda parte, si passa al foglio arch_fat e si trascrivono i dati, partendo dalla prima riga libera.

	A	B	C	D	E	F	G	H	I	J
1	Seri	NumFattur	DataFattur	CodClient	Articol	quantit	descrizior	iva%	Valore	
2	1	1	2/gen/2000	30	50	10	diario	4%	28.846	
3	1	1	2/gen/2000	30	60	30	compasso	20%	750.000	
4	1	1	2/gen/2000	30	70	40	album	10%	290.909	
5	1	2	2/gen/2000	10	50	10	diario	4%	28.846	
6	1	2	2/gen/2000	10	60	30	compasso	20%	750.000	
7	1	2	2/gen/2000	10	70	40	album	10%	290.909	
8										

```
Sub ArchiviaFattura()
' ArchiviaFattura Macro
' Macro registrata il 01/01/00 da Lucio

 Dim NumFattura, CodCliente As Long
 Dim DataFattura As Date
 Dim I, Righe As Integer
 Dim MatCodiciArticoli(20) As Long
 Dim MatQuantità(20), MatIva(20) As Single
 Dim MatDescrizioni(20) As String
 Dim MatValori(20) As Currency
 Dim ConPag As String

 Sheets("Fattura").Select
 'dati testata
 NumFattura = Cells(5, 3)
 DataFattura = Cells(6, 3)
 CodCliente = Cells(4, 7)

 Righe = 0
 'memorizza le righe di dettaglio
 For I = 11 To 38
    If Cells(I, 2) = "" Then
    Else
        Righe = Righe + 1
        MatCodiciArticoli(Righe) = Cells(I, 2)
        MatQuantità(Righe) = Cells(I, 3)
        MatDescrizioni(Righe) = Cells(I, 4)
        MatIva(Righe) = Cells(I, 7)
        MatValori(Righe) = Cells(I, 8)
                    -
    End If
 Next I

 'memorizza piede fattura
 ConPag = Cells(41, 2)

 'ricopia i dati nel foglio arch_fat
 Sheets("arch_fat").Select
 'seleziona l'ultima riga dell'archivio
 Cells(1, 1).Select
 Range("a1").End(xlDown).Select

 inizio = Selection.Row
 If inizio > 60000 Then inizio = 1
 For I = 1 To Righe
    Cells(inizio + I, 1) = 1
    Cells(inizio + I, 2) = NumFattura
    Cells(inizio + I, 3) = DataFattura
    Cells(inizio + I, 4) = CodCliente
    Cells(inizio + I, 5) = MatCodiciArticoli(I)
    Cells(inizio + I, 6) = MatQuantità(I)
    Cells(inizio + I, 7) = MatDescrizioni(I)
    Cells(inizio + I, 8) = MatIva(I)
    Cells(inizio + I, 9) = MatValori(I)
 Next I
 Cells(inizio + I - 1, 1).Select

End Sub
```

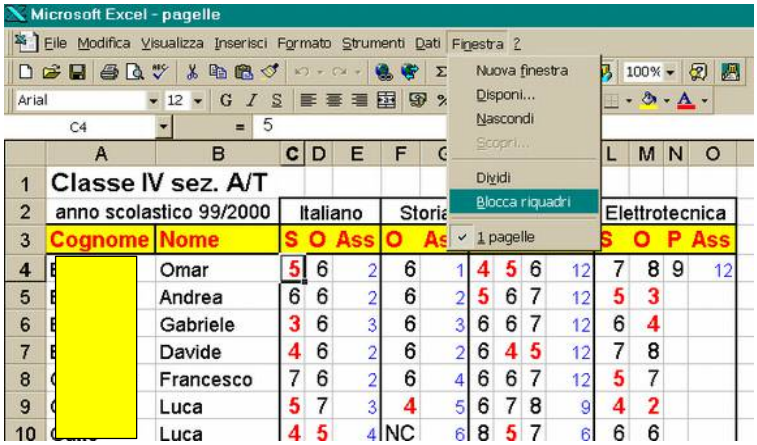
L'istruzione *Sheets("Fattura")* permette di passare al foglio *Fattura*.
Nei campi *NumFattura*, *DataFattura*, *CodCliente* vengono memorizzati i dati essenziali della testata (il numero della fattura, la data ed il codice cliente). Ragione sociale ed indirizzo non vengono memorizzati in quanto possono essere dedotti dal codice cliente, con l'ausilio dell'archivio *clienti*.
Il ciclo *For I = 11 to 38* permette di analizzare le eventuali righe di dettaglio, di contare quelle valide e di memorizzarne i dati nelle matrici *MatCodiciArticolo, MatQuantità, MatDescrizione, MatIva e MatValori*. Altri dati quali il prezzo o l'unità di misura possono essere dedotti dai precedenti o dall'archivio *articoli*.
Se il campo codice articolo (*Cells(I,2)*) è vuoto (""), la riga non contiene dati significativi e pertanto viene ignorata.
Alla fine del ciclo *For* il campo *Righe* contiene il numero di righe significative, da archiviare.
L'istruzione *Sheets("arch_fat").Select* permette di passare al foglio di lavoro *arch_fat*.
Il campo Inizio dovrà contenere il numero dell'ultima riga memorizzata in precedenza nell'archivio; le righe che si andranno ad aggiungere inizieranno dalla successiva.
L'istruzione Range("a1").End(xlDown).Select permette di selezionare l'ultima riga della colonna a contenente dei dati validi.
Il ciclo For I = 1 To Righe permette di memorizzare le "n=Righe" di dettaglio. Alla fine ci si posiziona nel primo campo della prima riga registrata.

SCRUTINI;

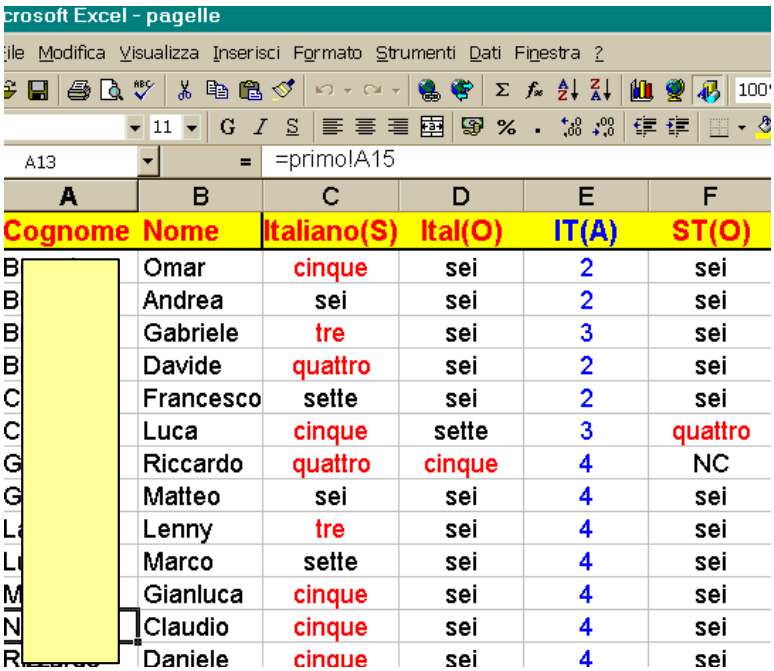
I nomi sono da ritenere di pura fantasia e così pure i voti, introdotti a caso.
Vogliamo ora vedere come automatizzare le operazioni di scrutinio:

- memorizzazione dei dati,
- stampa del tabellone,
- stampa delle pagelle.

In un primo foglio, indicato con il nome di “primo” memorizzeremo le proposte di voto e le assenze dei singoli allievi oltre ai loro nomi.



Per i voti si utilizzerà il formato condizione che consente di evidenziare in rosso le insufficienze ed in nero gli altri voti.
Per i numeri indicanti le assenze si sceglie il colore blu, in modo da rendere agevole l’introduzione dei dati.
Da notare che i nomi degli allievi devono essere scritti una sola volta, per tutti gli insegnanti.
Sempre al fine di rendere più agevole l’introduzione dei dati, vogliamo poter sempre vedere le prime due colonne con i nomi degli allievi e le prime righe con l’indicazione delle materie. Per far questo evidenziamo la cella C4 e scegliamo *Finestra/Blocca riquadri.*
Un secondo foglio “*Tab1*” consentirà di ottenere automaticamente il tabellone da esporre.
Da notare che Cognome si ottiene con la formula: “=primo!A3” che significa copia il contenuto della cella A3 del foglio “primo”. Stessa formula per tutti gli altri dati.
In pratica è sufficiente duplicare la formula in tutto il foglio di lavoro e poi cancellare le colonne corrispondenti alle assenze che non dovranno comparire nel tabellone. Scrivendo o modificando un dato del foglio “primo”, verrà automaticamente aggiornato il tabellone.
Un terzo foglio di lavoro dovrà contenere i dati che permettono di stampare le pagelle. Anche questo verrà ottenuto automaticamente dal primo foglio, sarà abbastanza simile al primo, con i voti espressi però in lettere.



La prima riga contenente le etichette permette di individuare i dati di ciascun record (con il termine record indichiamo l’insieme dei dati di un allievo, cioè di una riga).
I nomi degli studenti e le assenze vengono copiati automaticamente con la solita formula, dal “primo” foglio di lavoro.
Per la scrittura e traduzione dei voti in lettere si utilizza invece una ***function voti()*** scritta ad hoc.
Le celle contengono quindi formule del tipo:
“=voti(primo!C4)”
Vediamo ora la suddetta funzione:

```
Function voti(ByVal vinput As Variant) As String
'vinput non può essere dichiarato numerico,
'in quanto non sempre contiene un voto (è il caso ad esempio di NC
If IsNumeric(vinput) Then    'se la cella contiene un numero,
Else                         'cioè un voto
   voti = vinput             'nel caso ad esempio di NC
    Exit Function            'si esce dalla function
End If
Select Case vinput
Case Is = 1
    voti = "uno"
Case Is = 2
    voti = "due"
Case Is = 3
    voti = "tre"
Case Is = 4
    voti = "quattro"
Case Is = 5
    voti = "cinque"
Case Is = 6
    voti = "sei"
Case Is = 7
    voti = "sette"
'si prosegue fino al dieci
Case Else                    'negli altri casi
    voti = vinput
End Select                   'fine dell'istruzione Select
End Function
```

Per completare la stampa delle pagelle è sufficiente scrivere un documento con word, utilizzando ***Stampa Unione.***

Appendice:

Istruzioni di VB

Dichiarazione di variabili

Per dichiarare una variabile viene in genere utilizzata un'istruzione Dim. Un'istruzione di dichiarazione può essere posizionata all'interno di una routine per creare una variabile a livello di routine oppure può essere posizionata all'inizio di un modulo, nella sezione Dichiarazioni, per creare una variabile a livello di modulo.
Nell'esempio seguente viene creata la variabile strName e viene specificato il tipo di dati String.

Dim strName As String

Se questa istruzione compare in una routine, la variabile strName potrà essere utilizzata solo in tale routine. Se compare nella sezione Dichiarazioni di un modulo, la variabile strName sarà disponibile per tutte le routine del modulo, ma non per le routine di altri moduli del progetto. Per rendere questa variabile disponibile per tutte le routine del progetto, è necessario farla precedere dall'istruzione Public.

la sintassi semplificata è la seguente:
Dim *nomevariabile* As ***tipo***. . .

nomevariabileObbligatoria. Nome della variabile, espresso in base alle convenzioni di denominazione standard delle variabili.
tipo Facoltativa. Tipo di dati della variabile; può essere:
una costante booleana: *Boolean*
un numero intero (*Byte*, , *Integer*, *Long*)
un numero in virgola mobile (*Single*, *Double*)
un numero con quattro cifre decimali, usato per le valute (*Currency*)
una data (*Date*)
una stringa (*String*)
un oggetto (*Object*)
un tipo definito dall'utente
oppure può adattarsi al contenuto (*Variant* ;tipo di default)
esempio: Dim PiGreco as double
riserva spazio in memoria ad una variabile di nome PiGreco, dichiarata come numero in virgola mobile, di precisione doppia.
Si possono utilizzare anche vettori e matrici. In tal caso l'istruzione è la seguente:
Dim *nomevariabile(n,m)* As ***tipo***. . .
si dichiara una matrice di nome nomevariabile
n serve per riservare lo spazio ad n+1 righe
m riserva lo spazio in memoria per m+1 colonne
per un totale di (n+1)*(m+1) elementi
tipo (come in precedenza), si riferisce ai singoli elementi
esempio: Dim Parole(10) as String
dichiara un vettore colonna di 11 righe (da 0 ad 11) e ciascun elemento è una stringa

Suggerimenti:

Dichiarare sempre le variabili e preferibilmente all'inizio della sub o function.
Utilizzare Variant solo se necessario.
Dichiarare le variabili a livello di modulo o public solo se veramente indispensabile.

Riepilogo dei tipi di dati

La tabella che segue elenca i tipi di dati supportati e indica lo spazio su disco e l'intervallo valido per ciascun tipo di dati.

Tipo di dati			Spazio su disco	Intervallo
Numeri	interi			
		Byte	1 byte	Da 0 a 255
		Integer	2 byte	Da -32.768 a 32.767
		Long (intero lungo)	4 byte	Da -2.147.483.648 a 2.147.483.6477
	Virgola mobile	**Single** (virgola mobile a precisione semplice)	4 byte	Da -3,402823E38 a -1,401298E-45 per valori negativi; da 1,401298E-45 a 3,402823E38 per valori positivi
		Double(virgola mobile a precisione doppia)	8 byte	-4,94065645841247E-324 per valori negativi; da 4,94065645841247E-324 a 1,79769313486232E308 per valori positivi.
	valuta	**Currency** (intero diviso)	8 byte	Da -922.337.203.685.477,5808 a 922.337.203.685.477,5807
		Decimal	14 byte	+/-79.228.162.514.264.337.593.543.950.335 senza virgola; +/-7,9228162514264337593543950335 con 28 decimali; il numero minore diverso da zero è +/-0,0000000000000000000000000001
Boolean		**Boolean**	2 byte	True o False
Data		**Date**	8 byte	Dall'1 gennaio 100 al 31 dicembre 9999
Oggetto		**Object**	4 byte	Qualsiasi riferimento Object
stringa		**String**(lunghezza variabile)	10 byte + lunghezza stringa	Da 0 a circa 2 miliardi
		String(lunghezza fissa)	Lunghezza stringa	Da 1 a circa 65.400
Adattabile al contenuto		**Variant**(con numeri)	16 byte	Qualsiasi valore numerico fino all'intervallo di un Double
		Variant(con caratteri)	22 byte + lunghezza stringa	Stesso intervallo di String a lunghezza variabile
		Definito dall'utente (utilizzando Type)	Numero richiesto dagli elementi	L'intervallo di ciascun elemento è identico a quello del relativo tipo di dati sopraelencato.

Nota Le **matrici** di qualsiasi tipo di dati richiedono 20 byte di memoria più quattro byte per ogni dimensione della matrice più il numero di byte occupati dai dati stessi. La memoria occupata dai dati può essere calcolata moltiplicando il numero di elementi dei dati per le dimensioni di ciascun elemento. I dati in una matrice unidimensionale composti da quattro elementi di dati Integer di due byte ciascuno, ad esempio, occupano otto byte. Gli otto byte richiesti dai dati più i 24 byte di overhead portano a 32 byte la memoria complessiva richiesta per la matrice.

Un tipo di dati Variant contenente una matrice richiede 12 byte in più rispetto al numero di byte richiesti dalla matrice da sola.

Sub: Dichiara il nome, gli argomenti e il codice che costituiscono il corpo di una routine Sub:
End Sub: ultima istruzione di una Sub
sintassi semplificata:
Sub nomesub()
…istruzioni
…
End Sub
Suggerimenti:
aggiungere qualche riga di commento che spieghi quello che fa la sub, prima delle istruzioni

Function: Dichiara il nome, gli argomenti e il codice che costituiscono il corpo di una routine function:
End Function ultima istruzione di una function
sintassi semplificata:
Function nomefunction*(*argomento1 *as tipo1,* argomento2 *as tipo2) as tipo3*
…istruzioni
…
End Function
Si noti che gli argomenti sono separati da virgole e possono essere 0, 1, 2, 3 ecc. ecc.
Argomento1, argomento2 **non devono** essere dichiarati un'altra volta all'interno della function mediante l'istruzione Dim.
esempio di una Function:
Function AreaCerchio(ByVal raggio as single) as single
AreaCerchio=raggio ^2 * 3.14
end Function

Suggerimenti:
aggiungere qualche riga di commento che spieghi quello che fa la Function, prima delle istruzioni, in particolare indicare indicare quali sono i dati di input e quelli di output.
All'interno della **function** dovrebbe esserci almeno un'istruzione del tipo:
nomefunction = qualcosa.
è preferibile passare gli argomenti per valore pittosto che per indirizzo; davanti ad ogni argomento aggiungere pertanto la parola chiave ByVal; l'alternativa (per default) è ByRef .

Parole chiave relative agli operatori	
Aritmetici.	**^, – , *, /, \, Mod, +**
Di confronto.	**=, <>, <, >, <=, >=, …**
Logici.	**Not, And, Or, …..**
Per concatenare le stringhe di 2 espressioni	**&**
Funzioni	
Riguardanti la data	
Ottenere la data o l'ora corrente.	**Date, Now, ….**
funzioni aritmetiche	
Funzioni trigonometriche.	**Atn, Cos, Sin, Tan**
Calcoli generici.	**Exp, Log, Sqr**
Generare numeri casuali.	**Randomize, Rnd**
Ottenere un valore assoluto.	**Abs**
Ottenere il segno di un'espressione.	**Sgn**

Eseguire conversioni numeriche (restituiscono la parte intera di un numero).	**Fix, Int**
Controllo del flusso	
Salto incondizionato	**GoTo**
Uscire dal programma	**Exit**
Ciclo.	**Do...Loop,**
	For...Next,
	For Each...Next,
Operare scelte.	**If...Then...Else, End If**
	Select Case,
Utilizzare routine.	**Function, Sub**
trattamento stringhe	
Confrontare due stringhe.	**StrComp**
Convertire stringhe.	**StrConv**
Convertire in minuscole o in maiuscole.	**Format, LCase, UCase**
Creare una stringa di caratteri ripetuti.	**Space, String**
Trovare la lunghezza di una stringa.	**Len**
Formattare una stringa.	**Format**
Allineare una stringa.	**LSet, RSet**
Manipolare stringhe.	**InStr, Left, LTrim, Mid, Right, RTrim, Trim**
Varie	
Trattamento colori	**RGB**
Fare emettere un segnale acustico al PC.	**Beep**
Visualizzare un messaggio (output)	**MsgBox**
Immettere un dato. (input)	**InputBox**

Funzione **InputBox**

Visualizza un messaggio in una finestra di dialogo, attendendo che l'utente immetta del testo o scelga un pulsante, quindi restituisce un valore String
che include il contenuto della casella di testo.

Sintassi

InputBox(**prompt**[, title] [, default] [, xpos] [, ypos] [, helpfile, context])

La sintassi della funzione ***InputBox*** è composta dai seguenti argomenti predefiniti:

prompt Obbligatoria. Espressione stringa che costituisce il messaggio visualizzato nella finestra di dialogo. La lunghezza massima di prompt è di circa 1024 caratteri e dipende dalla larghezza dei caratteri utilizzati. Se prompt è suddiviso su più righe, è possibile includere, tra ciascuna coppia di righe, un ritorno a capo ***(Chr(13)),*** un carattere di avanzamento riga (***Chr(10))*** o una sequenza ritorno a capo-avanzamento riga (Chr(13) & Chr(10)).

Osservazioni

Se l'utente sceglie OK o preme INVIO, la funzione ***InputBox*** restituirà il contenuto della casella di testo. Se l'utente sceglie il pulsante Annulla, la funzione restituirà una stringa di lunghezza zero ("").

esempio:

DatoDiInput = InpuBox(prompt:="Inserisce il tuo nome"

==========

Funzione **MsgBox**

Visualizza un messaggio in una finestra di dialogo e attende che l'utente scelga un pulsante, quindi restituisce un valore Integer che indica quale pulsante l'utente ha scelto.

Sintassi:

MsgBox(**prompt[, buttons] [, title] [, helpfile, context**])

La sintassi della funzione MsgBox è composta dai seguenti argomenti predefiniti:

prompt **Obbligatoria**. Espressione stringa che costituisce il messaggio visualizzato nella finestra di dialogo. La lunghezza massima di prompt è di circa 1024 caratteri e dipende dalla

larghezza dei caratteri utilizzati. Se prompt è suddiviso su più righe, è possibile includere, tra ciascuna coppia di righe, un ritorno a capo (Chr(13)), un carattere di avanzamento riga (Chr(10)) o una sequenza ritorno a capo-avanzamento riga (Chr(13) & Chr(10)).

buttons Facoltativa. Espressione numerica che indica la somma dei valori che specificano il numero e il tipo di pulsante da visualizzare, lo stile di icona da utilizzare, il tipo di pulsante da utilizzare come impostazione predefinita e la modalità della finestra di messaggio. Se omesso il valore predefinito di buttons è 0.

Impostazioni

Alcune delle possibili impostazioni dell'argomento buttons sono:

Costante	Valore	Descrizione
vbOKOnly	0	Visualizza solo il pulsante OK.
VbOKCancel	1	Visualizza i pulsanti OK e Annulla.
VbCritical	16	Visualizza l'icona di messaggio critico.
VbQuestion	32	Visualizza l'icona di richiesta di avviso.
VbExclamation	48	Visualizza l'icona di messaggio di avviso.

esempio: MsgBox prompt: = "ciao", buttons = VbExclamation

Funzione RGB

Restituisce un numero intero Long che rappresenta un valore di colore RGB.

Sintassi

RGB(red, green, blue)

La sintassi della funzione RGB è composta dai seguenti argomenti predefiniti:

red Obbligatoria; Variant (Integer). Numero compreso nell'intervallo tra 0 e 255 inclusi che rappresenta il componente rosso del colore.

green Obbligatoria; Variant (Integer). Numero compreso nell'intervallo tra 0 e 255 inclusi che rappresenta il componente verde del colore.

blue Obbligatoria; Variant (Integer). Numero compreso nell'intervallo tra 0 e 255 inclusi che rappresenta il componente blu del colore.

Osservazioni

Le specifiche di colore ammesse per i metodi e le proprietà delle applicazioni sono rappresentate da un numero che rappresenta un valore di colore RGB. Il valore di colore RGB specifica l'intensità relativa dei colori rosso, verde e blu che uniti producono un colore specifico.

Il valore per qualsiasi argomento della funzione RGB superiore a 255 verrà considerato uguale a 255.

In tabella vengono indicati alcuni colori standard e i relativi valori di rosso, verde e blu:

Colore	Valore rosso	Valore verde	Valore blu
Nero	0	0	0
Blu	0	0	255
Verde	0	255	0
Azzurro	0	255	255
Rosso	255	0	0
Fucsia	255	0	255
Giallo	255	255	0
Bianco	255	255	255