```
/*
  Herzlich willkommen,
  in dieser dritten Einführungsintro werden Sie einige fortgeschrittene
  und mächtige Funktionalitäten in Gess Q. kennen lernen.

  Nachdem Sie dieses Skript durchgearbeitet haben können Sie:
  - Labels kopieren
  - Mehrere Fragen in einem Screen unterbringen
  - Asserts verwenden
  - Makros verwenden
  - Groups im Rahmen von Textersatz und Restricts einsetzen
*/


// - Grundeinstellungen importieren -
#include "formats.q"
// - Grafische Buttons verwenden -
#include "gbuttons.q"


// ---------------------------------------------------------------
// ---------- Inhalte der Umfrage --------------------------------
// ---------------------------------------------------------------
/*
  Also fangen wir an...

  mit einer kurzen Begrüßung, die abhängig vom angegebenen Geschlecht
  variieren soll:
*/

singleq frage1;
text="Bitte geben Sie Ihr Geschlecht an.";
labels=
1 "männlich"
2 "weiblich"
;

HTML{ backButton = yes; };


group begruessung;
labels=
1 "Sehr geehrter Mister X ..." (frage1 eq 1)
2 "Sehr geehrte Misses Y ..." (frage1 eq 2)
;

textq frage2;
text="
  @insert(begruessung)<br>
  Wir freuen uns, dass Sie an diesem Tutorial teilnehmen.
";

/*
  Hierzu wird eine "group" verwendet. Eine group ist kein Fragetyp,
  sondern ein Text- und Strukturelement. Wie Fragetypen enthält es
  jedoch labels gefolgt von einer Bedingung in runden Klammern. Die
  Bedingung kann in Form eines logischen Ausdrucks (wie bei Filtern)
  angegeben werden.

  Hier wird die group in einfacher Form verwendet. In geschweiften
  Klammern gibt sie konditional einen Text wieder. Eines der Labels
  muss hierzu einen wahren logischen Ausdruck besitzen.

  Etwas vorausgreifend: Sie dient ansonsten hauptsächlich als Vorlage
  für restrict-Anweisungen.
*/

/*
  machen wir weiter...

  ... mit einer MultiQ, die 20 Labels enthält. Mit den angeklickten
  Labels soll anschließend weiter gearbeitet werden. Mit dem Befehl
  'maxLabelsPerCol' werden die 20 Labels auf 2 Spalten verteilt.
  Spaßeshalber randomisieren wird die ersten 10 Label.

  In Tutorial 2 wurden die Labelzeilen wiederholt und einzeln gefiltert
  oder komplett mit dem restrict-Befehl. Das geht noch eleganter:
*/
```

```
HTML{ maxLabelsPerCol = 10; };
multiq frage3;
text="Welche der folgenden Produkte kennen Sie?";
title="Mehrfachnennung möglich.";
labels=
1  "Produkt 1 " random
2  "Produkt 2 " random
3  "Produkt 3 " random
4  "Produkt 4 " random
5  "Produkt 5 " random
6  "Produkt 6 " random
7  "Produkt 7 " random
8  "Produkt 8 " random
9  "Produkt 9 " random
10 "Produkt 10" random
11 "Produkt 11"
12 "Produkt 12"
13 "Produkt 13"
14 "Produkt 14"
15 "Produkt 15"
16 "Produkt 16"
17 "Produkt 17"
18 "Produkt 18"
19 "Produkt 19"
20 "Produkt 20"
;
assert (num(frage3) ge 5) "Sie müssen mindestens 5 Produkte kennen, um fortzufahren.";
// assert (num(frage3) ge 5) "Sie müssen mindestens 5 Produkte kennen, ..." exit 200;


/*
  Neu an dieser Stelle ist das assert, welches an 'frage3'
  gehängt wurde. Es überprüft eine bestimmte Bedingung nach dem
  Klick auf weiter.

  Dem Schlüsselwort 'assert' folgt ein logischer Ausdruck (wie bei Filtern).
  Anschließend wird ein Fehlertext und ein optionaler Abbruchcode
  (siehe auskommentierte assert-Zeile) angegeben.
  Gibt es einen Abbruchcode, wird bei Verstoß gegen die Bedingung das Interview
  an dieser Stelle mit dem angegebenen Code beendet. Gibt es ihn nicht, wird bei
  der aktuellen Frage verweilt und ein Fehlertext ausgegeben.

  Im Beispiel wird überprüft, ob mindestens 5 Produkte bekannt sind.

  Nun soll mit den bekannten Produkten weitergearbeitet werden:
*/

singleq frage4;
text="Welches Produkt davon nutzen Sie am meisten?";
labels copy frage3;
restrict=frage3;

/*
  Statt dem labels=... -Listing können mit "labels copy QNAME;" die
  Labels einer anderen Frage samt ihrer Parameter übernommen werden.
  Zusammen mit dem bekannten restrict werden die zuvor gewählten
  Produkte erneut im Kontext einer SingleQ wieder aufgegriffen.

  Ähnliche Dinge (und noch mehr) lassen sich mit Makros realisieren.
  Diese werden im Folgenden grundsätzlich vorgestellt.
  Makros sind in erster Linie nur Textersatz für sich ähnelnde
  Skriptabschnitte und können viel Tipparbeit sparen.
  Sie seien aber bedacht eingesetzt. Denn zu viele Makros können zu
  schlechterer Leserlichkeit führen.

  Es sollen 5 (fast) gleiche Fragen zu 5 unterschiedlichen Produkten
  gestellt werden:
*/

#macro makro_vorlage
singleq frage$1;
text="
  Haben Sie <span style='color:#$3;'>$2</span> schon einmal gekauft?
";
labels=
1 "Ja"
2 "Nein"
;
#endmacro
```

```
/*
  Es wurde ein Makro mit dem Namen 'makro_vorlage' definiert.
  Bis hierhin ist es nur eine Vorlage. Es hat noch keinerlei Effekt.
  Zwischen #macro und #endmacro befindet sich die Frage, die wir
  mehrfach reproduzieren wollen. Es handelt sich um ganz normales
  Q. Skript - bis auf eine Kleinigkeit:

  Dollarzeichen gefolgt von einer Zahl $1 $2 $3

  Hierbei handelt es sich um Platzhalter, von denen beliebig viele in
  den Code eingebaut werden können.

  Instanziieren wir mit dieser Vorlage nun eine echte SingleQ:
*/

&makro_vorlage;$1="5";$2="Produkt A";$3="990000";

/*
  Mit einem UND-Zeichen & gefolgt vom Makronamen wird nun eine echte
  Instanz der Vorlage erzeugt:
  - &makro_vorlage;
  Damit die Platzhalter $1 $2 $3 nicht im Scriptcode bleiben muss für
  sie jeweils ein Ersatz angegeben werden.
  - $1="5"
  setzt an die Stelle des Platzhalters eine 5, usw.

  Der tatsächlich erzeugte Skriptcode sieht folgendermaßen aus:

  singleq frage5;
  text="
    Haben Sie <span style='color:#990000;'>Produkt A</span> schon einmal gekauft?
  ";
  labels=
  1 "Ja"
  2 "Nein"
  ;

  Nun werden 5 weitere Instanzen für die Produkte B,C,D,E mit unterschiedlichen
  Schriftfarben (HEX Angabe für $3) erzeugt und die entsprechenden fragen
  frage5 - frage 9 in einem Block zusammen gefasst:
*/

&makro_vorlage;$1="6";$2="Produkt B";$3="009900";
&makro_vorlage;$1="7";$2="Produkt C";$3="000099";
&makro_vorlage;$1="8";$2="Produkt D";$3="999900";
&makro_vorlage;$1="9";$2="Produkt E";$3="009999";

block makro_block1 = ( frage5 frage6 frage7 frage8 frage9 );

/*
  An welcher Stelle und in welchem Umfang Makros eingesetzt werden ist
  grundsätzlich Ihnen überlassen. So kann die Produktliste aus Frage3
  auch mit einem Makro abgebildet werden:
*/

#macro macro_labels_frage3
1  "Produkt 1 " $1 $2
2  "Produkt 2 " $1    $3
3  "Produkt 3 " $1 $2
4  "Produkt 4 " $1    $3
5  "Produkt 5 " $1 $2
6  "Produkt 6 " $1    $3
7  "Produkt 7 " $1 $2
8  "Produkt 8 " $1    $3
9  "Produkt 9 " $1 $2
10 "Produkt 10" $1    $3
11 "Produkt 11"    $2
12 "Produkt 12"       $3
13 "Produkt 13"    $2
14 "Produkt 14"       $3
15 "Produkt 15"    $2
16 "Produkt 16"       $3
17 "Produkt 17"    $2
18 "Produkt 18"       $3
19 "Produkt 19"    $2
20 "Produkt 20"       $3
#endmacro
```

```
/* Zusätzlich wurde diese Liste nun noch mit 3 Parametern versehen:
 * $1 erlaubt, die Randomisierung der ersten 10 Fragen einzustellen
 * $2 ermöglicht es, die Produkte 1,3,5,7... zu filtern
 * $3 ermöglicht es, die Produkte 2,4,6,8... zu filtern
 *
 * Aber zuerst soll frage3 identisch reproduziert werden:
*/

multiq frage3_2;
text="Welche der folgenden Produkte kennen Sie?";
title="Mehrfachnennung möglich.";
labels=
&macro_labels_frage3;$1="random";$2="";$3="";
;

/*
  Die Randomisierung stellen wir mit Parameter $1="random"; her,
  die übrigen Parameter werden nicht benötigt und werden leer gelassen.

  Auf diese Art und Weise könnten z.B. die Labellisten bei sich
  sich wiederholenden Umfragen über Makros in einer separaten Datei
  gepflegt werden.

  Ferner lassen sich mit dem Makro macro_labels_frage3 Gruppen definieren,
  mit denen auf bestimmte Produktuntergruppen gefiltert werden kann.
  Es werden 2 Gruppen angelegt, mit denen die Produkte mit ungeraden/geraden
  Produktnummern gefiltert werden können:
*/

group grp_produkte_ungerade;
labels=
&macro_labels_frage3;$1="";$2="(true)";$3="(false)";
;
group grp_produkte_gerade;
labels=
&macro_labels_frage3;$1="";$2="(false)";$3="(true)";
;

/*
  In diesem Fall wird der Parameter $1 nicht benötigt und leer gelassen,
  $2="(true)" und $3="(false)" in group grp_produkte_ungerade filtert die
  'ungeraden Produkte', anders herum filtert group grp_produkte_gerade die
  'geraden Produkte'. Das Prinzip ist identisch zur verwendeten Gruppe für
  den Begrüßungstext.

  Diese groups können nun als Parameter für die restrict-Anweisung verwendet
  werden. Die Randomisierung über $1 in macro_labels_frage3 lassen ist einmal
  aktiv und einmal ausgelassen.
*/

multiq frage3_2_ungerade;
text="Welche der folgenden Produkte kennen Sie?";
title="Mehrfachnennung möglich.";
labels=
&macro_labels_frage3;$1="";$2="";$3="";
; restrict = grp_produkte_ungerade;


multiq frage3_2_gerade;
text="Welche der folgenden Produkte kennen Sie?";
title="Mehrfachnennung möglich.";
labels=
&macro_labels_frage3;$1="random";$2="";$3="";
; restrict = grp_produkte_gerade;

block makro_block2 = (frage3_2 frage3_2_ungerade frage3_2_gerade);


block main = ( frage1 frage2 frage3 frage4 makro_block1 makro_block2 );
```