REDCap
Branching logic Class

November 9, 2017

REDCap

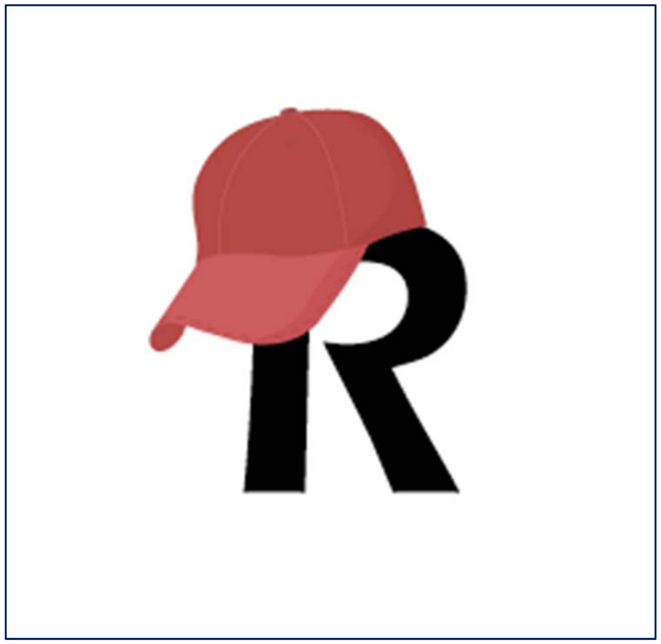
Research Electronic Data Capture

ITHS | Institute of Translational Health Sciences

Accelerating Research. Improving Health.

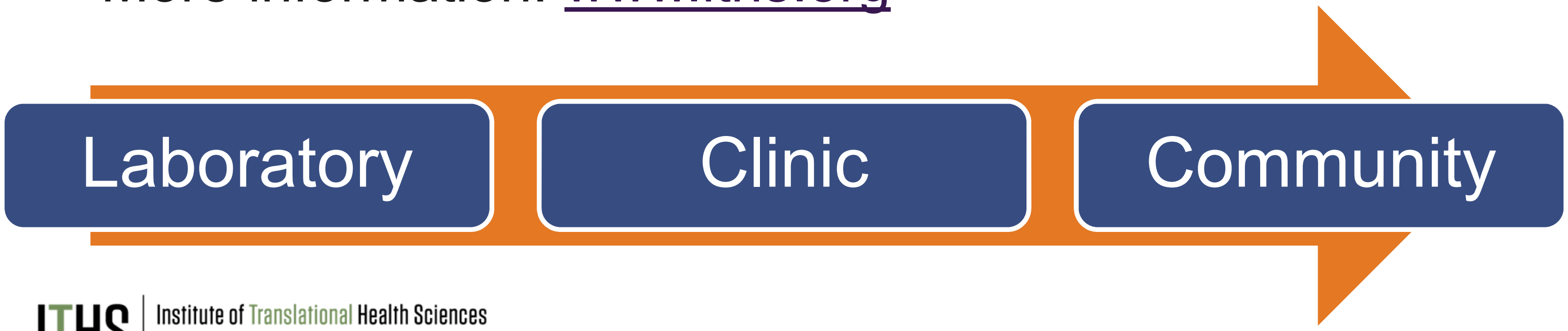
Learning objectives

- Branching logic basics
- Simple statements
- Complex statements
- Special functions
- Longitudinal branching logic
- Interplay with action tags
- Creative uses of branching logic

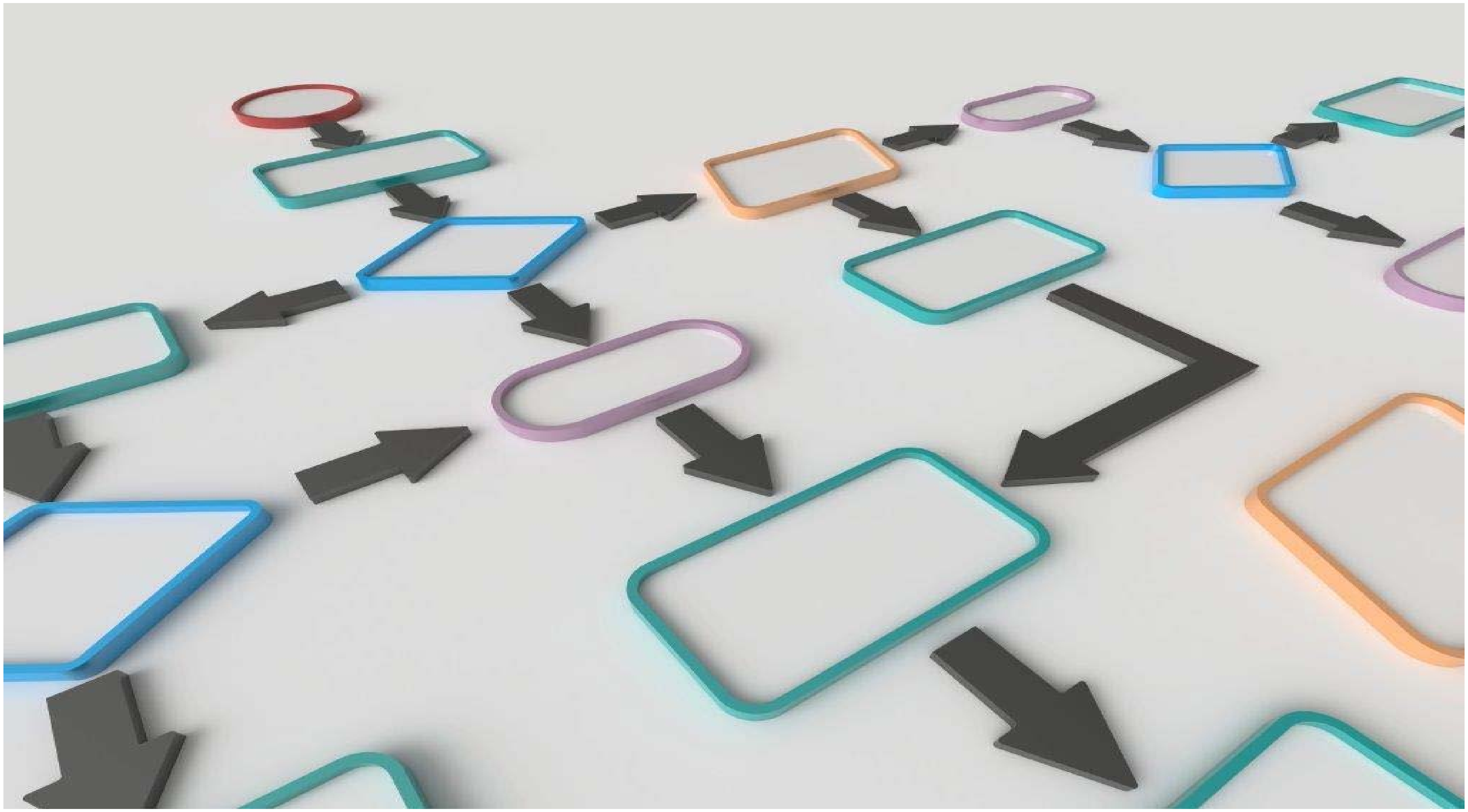


ITHS’ Focus

- Speeding science to clinical practice for the benefit of patients and communities.
- Promotes translation of scientific discovery by:
  - ❑ Fostering innovative research
  - ❑ Cultivating multi-disciplinary partnerships
  - ❑ Training the next generation of researchers
- More information: www.iths.org

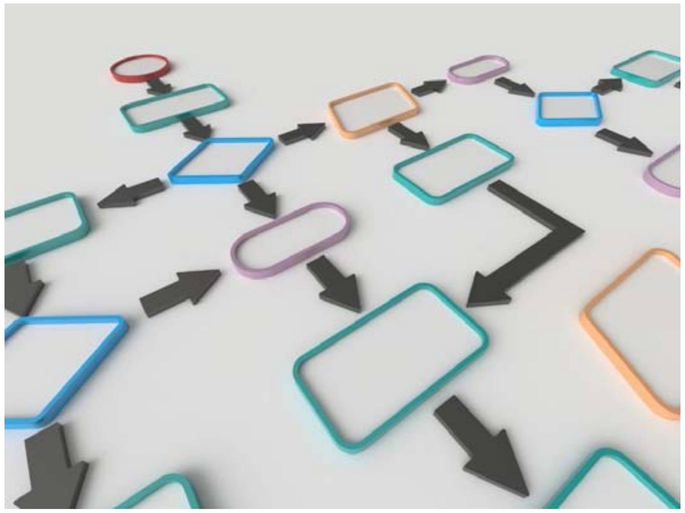


BRANCHING LOGIC



Institute of Translational Health Sciences
Accelerating Research. Improving Health.

What’s Branching Logic?



Adds flexibility to your instruments

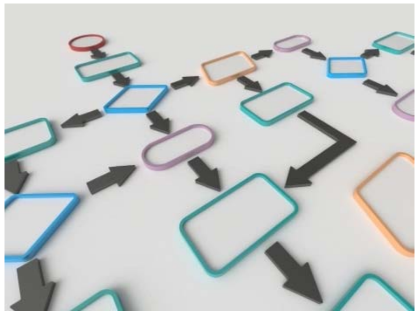
- The art of hiding or showing fields
- Based on previously entered values
- Limited to a single project
- Drag and Drop Method:
  - The “easy” way
  - Reduced flexibility
- Advanced Syntax Method:
  - The “hard” way
  - Programming experience helps
  - Allows you to get creative
  - Can be used both in the online interface and in the data dictionary

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Reversing your thought process

- Logic located in the "source" question
- Directs you to "skip" to a question down the line
- Very linear
- Hard to account for complex logic
- Mostly found on paper forms

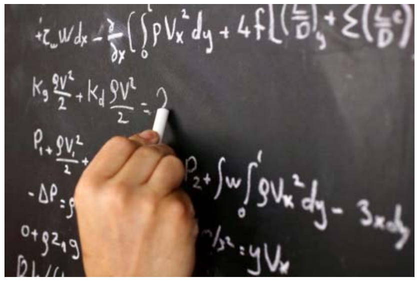
Classic skip logic



- Logic located in the "destination" question
- Hides or shows the question
- Allows for multiple logic pathways (e.g. Inclusion criteria)
- Can get very complex
- Extensive used in REDCap

Branching logic

The language of logic in REDCap



- And Statement
  - **and**
- Or statement
  - **or**
- Equals statement
  - **=**
- Not statement
  - **<>**

Logic

- Standard math
  - **+, -, /, ***
- Comparing
  - **>, <, >=, <=**
- Order of operation
  - **Parentheses ()**

Math

Building a basic logic statement

- End result always needs to be a “true” or a “false”
- Define the **variable**
  - Brackets (e.g. **[variable1]**)
  - Use the variable name instead of the field label
  - Brackets are also used for event definition (e.g. **[baseline_arm_1][variable1]**)
- Put in an **operator**
  - e.g. **= , <>, >=, <=, >, <**
- Declare you **comparison value**
  - Can be a ”hard” value like a number or a date
  - Can be another variable
  - Use of single quotes is optional
    - Double quotes allowed, not recommended

[age_of_child] >= ‘18’

Branching logic example 1
Simple statement (single/radio)

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements

Logic context

- You want to ask the question:
  Is the participant on Medicare?
  But this is only relevant for people over 65.

Needed elements

- Variable: [age]
- Operator: >=
- Comparison value: ‘65’

Branching logic statement

- **[age] >= ‘65’**

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 2

Simple statement (checkbox)

Basic statements

Simple (Single/Checkbox)

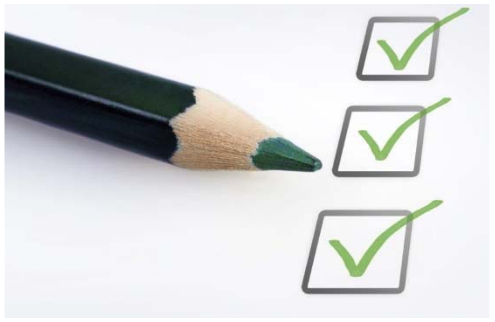
And

Or

Not

Empty

Complex statements



Logic context

- You want to ask the question:
  Did the participant get vaccinated for malaria?
  But this is only relevant for people who recently went to a country where malaria is prevalent .

Needed elements

- Variable: [country(3)]
- Operator: =
- Comparison value: ‘1’

Branching logic statement

- **[country(3)] = ‘1’**

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 3

And statement

Basic statements

- Simple (Single/Checkbox)
- **And**
- Or
- Not
- Empty

Complex statements

Logic context

- You want to ask the question:
  How did the medication affect your allergy symptoms?
  But this is only relevant for people who have allergy symptoms and take the medication.

Needed elements

- Variables: [allergy] and [symptoms]
- Operator: =
- Comparison value: '1'

Branching logic statement

- **[allergy] = '1' and [symptoms] = '1'**

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 4
Or statement

Logic context

- You display a warning to warn for ineligibility when a participant is either a smoker or a drug user.

Needed elements

- Variables: [smoker] and [drugs]
- Operator: =
- Comparison value: ‘1’

Branching logic statement

- **[smoker] = ‘1’ or [drugs] = ‘1’**

Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 5
Not statement

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements

Logic context

- ► You want to ask the question:
  Have you ever had heart attack-like symptoms?
  But you only want to ask this when people have NOT had a heart attack before.

Needed elements

- ► Variable: [heart_attack]
- ► Operator: <>
- ► Comparison value: ‘1’

Branching logic statement

- ► **[heart_attack] <> ‘1’**

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 6

Empty statement

Logic context

- You want to display a warning when the date of birth field has not been filled out. But the warning needs to disappear if the date of birth field has a value in it.

Needed elements

- Variable: [dob]
- Operator: =
- Comparison value: '' *(two single quotes)*

Branching logic statement

- **[dob] = ''**

Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 7

Complex statement

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Logic context

- You want to display a question about mid life crisis, but only when the participant is outside of the standard mid life crisis age range and the date of birth field has been filled out.

Needed elements

- Variable: [age], [dob]
- Operator: >=, <=, <>
- Comparison value: ‘0’,’39’,’63’,’120’,’’ *(two single quotes)*

Branching logic statement

- **([age] >= ‘0’ and [age] <= ’39’ and [dob] <> ‘’) or ([age] >= ’63’ and [age] <= ‘120’ and [dob] <> ‘’)**
- **(([age] >= ‘0’ and [age] <= ’39’) or ([age] >= ’63’ and [age] <= ‘120’)) and [dob] <> ‘’**

Branching logic example 8
Date differential

Logic context

- You want to display a question about how somebody's heart attack affected their work life, but only if they had the heart attack when they were younger than 65.

Needed elements

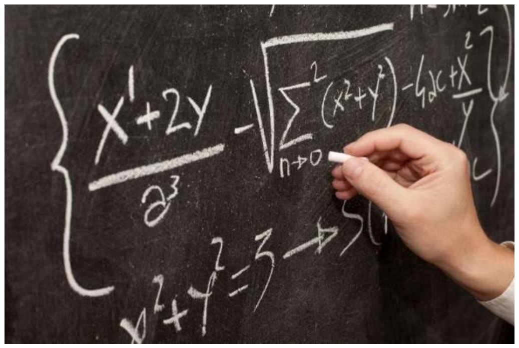
- Variable: [dob], [date_of_attack]
- Function: datediff([date1],[date2],"units","format")
- Operator: <=
- Comparison value: '65'

Branching logic statement

- **(datediff([dob],[date_of_attack],'y','mdy')) <'65'**

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Branching logic example 9
Sum statement



ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Logic context

- You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value.

Needed elements

- Variable: [depr1], [depr2], [depr3]
- Function: sum()
- Operator: >=
- Comparison value: ‘4’

Branching logic statement

- **(sum([depr1],[depr2],[depr3]))>=’4’**

Branching logic example 10
Contains statement

Logic context

- You want to ask for a survey respondents private email if they provide an university email address in their initial response.

Needed elements

- Variable: [email]
- Function: contains()
- Comparison value: '.edu'

Branching logic statement

- **contains([email],'.edu')**

Branching logic example 11

If statement

Logic context

- You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value. However, you've built in a "prefer not to answer" response for the first question that you coded as '99'. You need to filter out this response from your logic.

Needed elements

- Variable: [depr1], [depr2], [depr3]
- Function: if(), sum()
- Operator: >=,=
- Comparison value: '4','99','0'

Branching logic statement

- **(sum(**
  (if([depr1]='99','0',[depr1]))
  ,[depr2],[depr3]))>='4'

Branching logic example 12

Nested if statement

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Logic context

- You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value. However, you've built in a ”prefer not to answer” response for the first question that you coded as ’99’. You've also added an option for “unknown” (98). You need to filter out these responses from your logic.

Needed elements

- Variable: [depr1], [depr2], [depr3]
- Function: if(), sum()
- Operator: >=,=
- Comparison value: ‘4’,’99’,’98’,’0’

Branching logic statement

- **(sum(**
  (if([depr1]=‘99’,’0’,
  (if([depr1]=‘98’,’0’,[depr1]))))
  ,[depr2],[depr3]))>=’4’

Branching logic
More functions

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Location

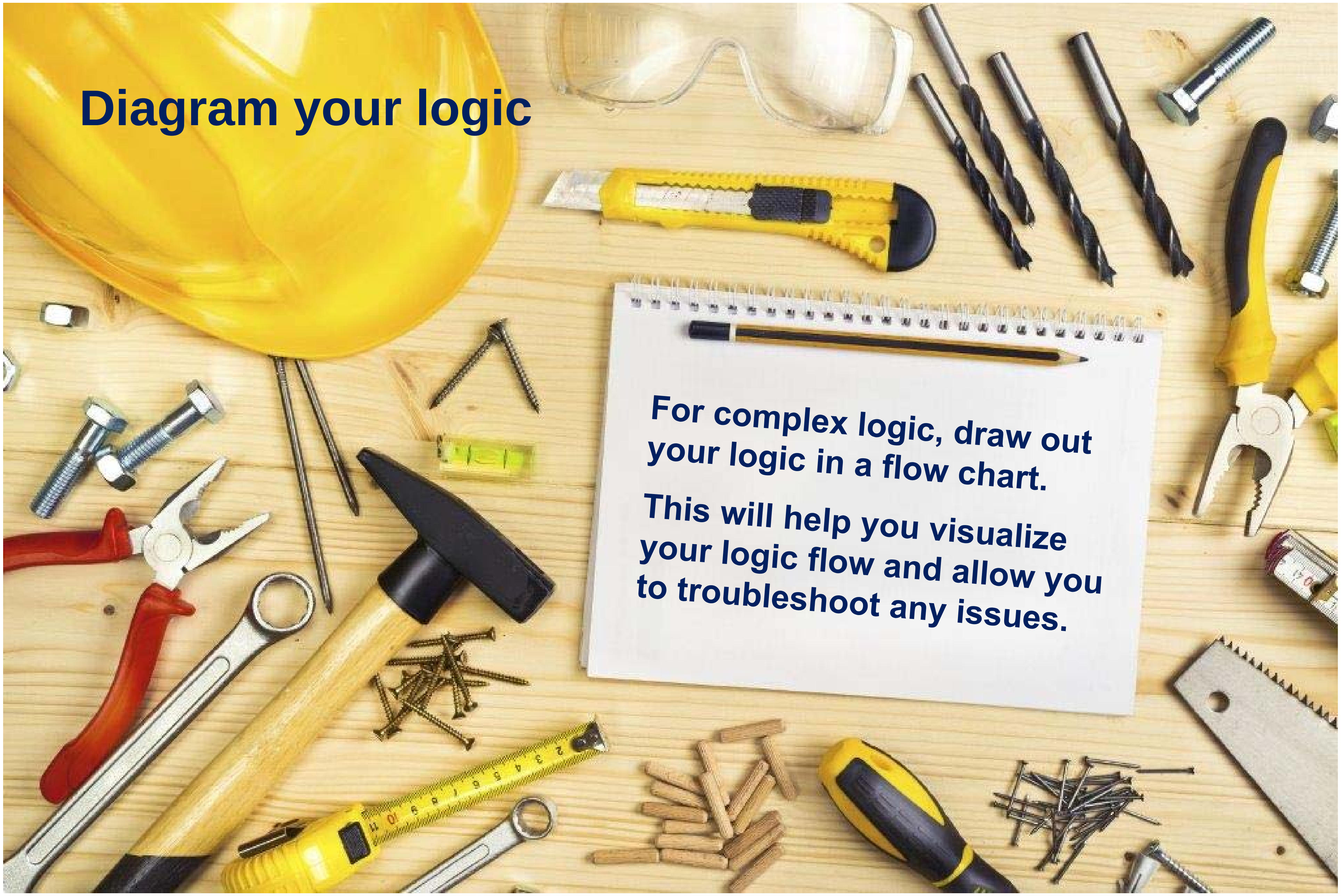
- For other complex functions look in the FAQ

FAQ section on functions

(Links to the ITHS REDCap installation)

- Examples:
  - Rounding
  - Square root
  - Mean
  - Median
  - Exponents
  - Minimum/Maximum
  - Standard deviation
  - Logarithm
  - Is a number?

Diagram your logic



For complex logic, draw out your logic in a flow chart.

This will help you visualize your logic flow and allow you to troubleshoot any issues.

Longitudinal Branching Logic

- **Classic Branching logic**
  - Define the **variable**
  - Put in an **operator**
  - Declare you **comparison value**

```
[age_of_child] >= ‘18’
```

- **Longitudinal Branching Logic**
  - Define the **event**
  - Define the **variable**
  - Put in an **operator**
  - Declare you **comparison value**

```
[baseline_arm_1][age_of_child] >= ‘18’
```

Interplay with action tags

Action tags usually "win out" over branching logic

- Action tags and branching logic can be used concurrently if desired.
- Most action tags do not affect branching logic
- Exceptions:
  - @HIDDEN
    Hides the field regardless of the logic result
  - @DEFAULT
    Will only prefill a field if the field is shown initially.
    If it's hidden with logic the @DEFAULT tag will not work.

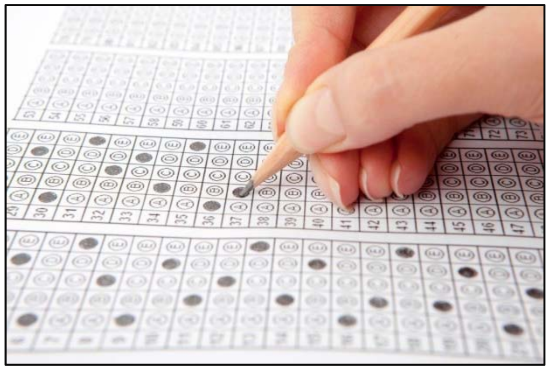
Creative Use: Cascading logic

Use cascading logic to simplify your logic

- Cascading logic can greatly simplify all your logic statements.
- Each statement would only need to look at it's presceding variable
- e.g. Cascading medication lists:

Variable	Cas. logic	Non cascading logic
rx1		
rx2	[rx1]<>""	[rx1]<>""
rx3	[rx2]<>""	[rx1]<>"" and [rx2]<>""
rx4	[rx3]<>""	[rx1]<>"" and [rx2]<>"" and [rx3]<>""

Creative Use: Score evalution



Combing logic with descriptive fields to show situational messages

- Adding logic to descriptive fields is a great way of communicating certain results
- e.g Scoring tool evaluation:
  - Calculated field that generates a score
  - Three descriptive fields:
    - Below average
    - Average
    - Above average
  - Example: https://is.gd/logicdemo

ITHS Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Thank You

Questions?

CONNECT WITH ITHS

www.iths.org

@ITHS_UW

/ithsuw

/InstituteofTranslationalHealthSciences

ITHS | Institute of Translational Health Sciences
Accelerating Research. Improving Health.

Visit ITHS.org to Become an ITHS Member

Join a unique catalyst that accelerates discoveries to practice.

Access

Members gain access the different research services, resources, and tools offered by ITHS, including the ITHS Research Navigator.

Education and Training

Members can access a variety of workforce development and mentoring programs and apply for formal training programs.

Funding

Members can apply for local and national pilot grants and other funding opportunities. ITHS also offers letters of support for grant submissions.

Collaboration

Members can connect with collaborators across the CTSA consortium.