Hello!

This program was designed to be used by behaviorist researchers to study treatment integrity as it is relevant to DRO (Differential reinforcement of other behaviors). It is meant to be flexible to the needs of the researchers, who are assumed to not be programmers themselves. Control of the research design is given to the researcher through an excel document which is referred to, and will from this point be referred to, as the "Script". The program is written using the library PsychoPy, more will be discussed on this later.

FOR THE RESEARCHER:

In this version of the program, there should be three files in the program. These three files need to be in the same folder for the program to work, because one of the files (the program) will look for the other two files (script and point sound). Let me tell you what you need to know about them.

(1) Combined.py: This is the actual program which is written in python (indicated by the .py file extension). This will be what you run when you want to run the program. Now, you'll notice that you can't just double click this to open it. There are a few ways in which it can be run, but what I think will be the most convenient for you is to download the standalone PsychoPy program, which can be found here https://github.com/psychopy/psychopy/releases . Now, the most recent version will likely work, but I have had some issues in which updating the version actually made the program not work. So, to be safe, you should scroll down on the releases and download the version “2023.1.3” which is the version I used to make it. When you download and install PsychoPy, you will be able to use the "PsychoPy Runner" to select and run the program. I am not sure if you will need to also download python, I’m pretty 99% sure psycopy will install python for you.

(2) Script.xlsx : This is the script, the part of the program that you will use to edit the order and type of phases of the program. Each row is a phase, with order starting from the top. Now, depending on the phase of the program, what you can put (and what you need to put) in the "Reinforcement\_Schedule", "Comission\_Integrity", "Omission Integrity", and "Reinforcment\_Variable" columns. I will tell you about what each phase is, and while doing this, I will explain the specifics of what each phase needs. What all phases need is a value in Order, which should always be equal to the excel row - 1 (which is to say it should just be 1,2,3,4..). It also needs a duration, which is in seconds. And obviously, each row will need a phase assigned to it. Now let me explain to you the different phases, keep in mind that for the program to read the excel document all values need to be used exactly as they are used below (i.e., program cannot read typos):

“Assessment”: This is the reinforcer assessment. In this phase, there are two buttons instead of one, a black target and a red distractor (which is always placed on extinction). The black target is the one augmented by the phase instructions. The assessment needs to have a reinforcement schedule of either “FI” or “FR”. It also needs to have a reinforcer variable, which augments the reinforcement schedule such that a schedule “FI” and a variable “3” means that the black target will be on an FI-3 schedule. This works similarly in the “BL” phase. Currently the program is written such that at the end of the assessment stage, the points reset back to 0.

“Break”: The break phase only needs to have the duration column filled in. The break phase works like this, text pops up on the screen letting the participant know that they have an optional break. To end their break early they must click an orange button, which shows up a few seconds after the break starts (just to avoid any accidental clicks). Additionally, a timer counts down how much time they have left on their break. When it reaches 0, it does not go to the next phase, rather prompts them to click the orange button to continue.

“BL”: The BL phase works similarly to the assessment phase, the only differences are that that there is no red button and that points don’t reset at the end.

“Intervention”: The Intervention phase is the DRO phase. This means that the “Reinforcement\_Schedule” has to be “DRO” for this phase, it is the only option. It also needs a “Reinforcer\_Variable” which represents the number of seconds without clicking that the participant must go to earn a non-error point. Now, as the program is set up to test the integrity of DRO, we also need to have values in “Commission\_Integrity” and “Ommission\_Integrity”. These values need to range from 100-0. Let me explain how these numbers work. An integrity of 100 means there is no chance for error, and a 0 means that it will always error. The Commission Integrity determines the chance a click on the button will result in a point being delivered (which is an error, as points should not be awarded from clicks during DRO). The Ommission Integrity determines the chance that a point will fail to be rewarded from the successful completion of a DRO cycle.

That should be all the information you will need to use the script.

(3) Point.wav: This is the sound file that plays when a point is earned. If you wish to change this file, just make sure that the file you use to replace it with is called “Point.wav” so the computer can read it. No garuntees that changing it will not cause a bug.

Data:  
Now, in addition to these three files, the program will produce two file folders. One will be called “data”, this is produced by PsychoPy and you can ignore it. The folder you care about is “participant data”. Each folder is marked by each session’s participant id. In it will be four files. Two of them will have the file extension “.psydat”, they can be ignored. You care about the two excel files “RawData.csv” and “ClicksData.csv”. All of the information that is in Clicks Data is in Raw Data. Since the dependent variable of this study is clicks, it’s given a folder all to its own.

RawData.csv: This includes a record of all events that go on in the experiment. This can be thought of as a sort of reliability check because it will give you more information like when phase changes occur, when errors occur, when DRO points are awarded, and current total points. You will certainly need to use it to check how often errors occurred in the experiment.

ClicksData.csv: This includes a record of all clicks that have gone in the experiment. Each row is a click, and includes information on whether or not that click awarded a point, what sort of point was awarded (I.e., was it a commission error or was it a FI 3). Additionally, for the assessment phase, it tells you what was clicked on.

NOTE: When you run the program, you will be prompted with a textbox that says participant ID and is filled in with a random number. Even though you can edit the participant ID, it doesn't change the number in the data and just uses the original random number. So just record the number you are given to keep track of which participant is which. If you ever get confused remember you can check the time created of the files which can be a good clue to what data set is what, I have to do this often.

FOR THE PROGRAMMER:

About my programming knowledge:

Just to warn you, I am

PsychoPy: The program is written using PsychoPy. If you google PsycoPy it will say it is a “software package”, at this point I am only using it as a library/framework (these three things may be interchangeable terms, I’m not sure). Anyways, let me explain how I have come to be using PsycoPy, because I think it will help you understand (1) why the code is written the way it is and (2) if you want to continue using PsycoPy.

Originally, PsycoPy was chosen because it was free, and comes with a no-code (code-optional?) builder, and has lots of built in functionality that is useful for experiments. Now, I worked with the no-code builder for a few weeks, however it was not flexible enough to actually do what we needed, so I eventually transitioned to just coding it. However, the PsycoPy builder has a like “builder to python” translator, so I didn’t completely start over, I just moved translated it all to python and began working from there. This is to say, I didn’t write all of the code, mainly I wrote lines 40-400, while changing lines throughout the whole code. Anyways, the PsycoPy builder is kind of useful but it’s also kind of a gimmick, ultimately the goal of it being a tool researchers could use to change the program after I left did not work out. Anyways, psychopy has succeeded on the other front, it has provided a lot of useful things automatically. I like the way the data taking works, including how it’s stored. It’s built in UI also certainly gets the job done. At the same time, it’s taken me quite a while to understand how their code works, I wonder if it would have been faster to just write everything from scratch.   
Program Structure:

So PsycoPy has what they call routines, which exist inside of a loop. Each routine exists without a “While continueRoutine” where continue routine is a Boolean. The first routine in this program is the “Initialize” routine. It initializes pretty much all the functions for the experiment and prompts the user to “click to continue”. The main routine is next. It encompasses the experiment, and can be controlled by the researcher using the “Script.xlsx” file. Each row of this file is one loop, each time the routine has been going for the duration designated for that row, the experiment moves on to the next row. In the section above on this file, more is explained about the specific conditions. Anyways, this routine really consists of three function calls. One which calls the function “MouseClicked()” anytime the mouse is clicked, and two others which are called every frame, “RoutineEnder()” and “EachFrameChecker()”. These are called at 844-852. Let me quickly summarize these three functions

“MouseClicked()”: This function does a few things, it takes data every time the mouse is clicked, it assigns appropriate points based on the specific reinforcement contingencies in place, and it causes the button to move to a new random spot and become invisible (and effectively unclickable) for a fraction of a second. It uses a variety of helper functions to do this, which are commented out in the code. This is also used in the break phase of the experiment,

“RoutineEnder()” is called every frame. It ends the routine when the duration of the phase has passed. For reason’s beyond my understanding, (probably has something to do with the like globalility/locality of the variable “continueRoutine”), the continueRoutine Boolean can not be changed inside of this function, but only in the code of the routine loop. So, this returns a Boolean to let the loop know if it is time for the routine to end. This function also takes data on the phase ending.

“EachFrameChecker()” : is a function that causes the button to become clear again after the fraction of a second is up, it also is what monitors if the participant has not clicked for 3 (or however many seconds the DRO is set too) in the differential reinforcement condition.

On top of these three functions, the “break” phase and reinforcer “assessment” has it’s own function associated with it, “BreakClick()” which ends the break phase early and “RedClicked” which takes data when the red button is clicked, and resets the two button locations.

Anyways, there is more to the program, including more auxiliary functions, but these are the really important ones. I figure if you understand when these 3 main functions are called, you will be able to figure everything else out about the program. Will need to add comments.