Roteiro

- objetivos da aula:

* Desafio de programação: Implementar a tarefa de comprimento de dígitos (*digit span task*; Jacobs, 1887), com base na versão descrita por Johnson et al. (2014);
* **Digit span:** o número máximo de dígitos sequencialmente apresentados que podem ser confiavelmente recordados na ordem correta (Baddeley et al., 2020);
  + Exemplo 1: dígitos em uma mesma sequência **podem** se repetir;
  + Exemplo 2: dígitos em uma mesma sequência **não podem** se repetir;
  + Exemplo 3: dígitos devem ser lembrados na ordem inversa de apresentação (backward digit span). Dígitos em uma mesma sequência **não podem** se repetir;
* Exemplo 1—forward digit span (dígitos **podem** se repetir em uma mesma sequência):
  + **Instruções:** Usando um objeto do tipo **list** e a classe **visual.ButtonStim**, criamos as instruções que aparecerão em cada tela nas iterações de um loop. Os botões *Avançar* e *Retornar* deixarão as instruções esteticamente mais agradáveis;
  + **Loop aninhado em três níveis:** loop interno (**stimuli**) controlará a apresentação dos diferentes dígitos de uma sequência (tentativa). Uma tentativa será operacionalizada como o número total de dígitos apresentados nesse loop (**nReps = series\_len**, cujo valor variará entre 2 e 9). O loop intermediário (**span\_trials**) tem número fixo de repetições (**nReps = 2**). Ele é usado para modelar a tarefa de Johnson et al. (2014), que usaram duas repetições de cada comprimento de sequência (e.g., dois tentativas com 3 dígitos, duas com 4 dígitos, e assim por diante). Por fim, o loop externo (**trials**) controlará o valor atual da sequência de dígitos a ser apresentada ao participante (**series\_len**);
  + **Prompt:** rotina temporária, usada apenas para controlar o início de uma nova tentativa;
  + **Study:** código usa **numpy.random.randint()** para gerar números inteiros entre 0 e 9, que serão apresentados na tela. O valor gerado por este comando é monitorado a cada iteração, de modo a salvar a sequência apresentada ao participante, bem como para comparar essa sequência posteriormente com a resposta do participante;
  + **Recordação:** monitora as teclas que o participante pressionou, exibe na tela as teclas digitadas, salva as variáveis importantes no arquivo de saídas e monitora se o participante acertou pelo menos 1 das 2 tentativas de cada iteração do loop **span\_trials**. Caso o participante tenha acertado 0 de 2 tentativas, o loop **trials** é encerrado.
  + **Agradecimentos:** rotina apenas comunica ao participante que a tarefa chegou ao fim;
* Exemplo 2—forward digit span (dígitos **não podem** se repetir em uma mesma sequência):
  + Ao invés de usar **numpy.random.randint()**, aqui usamos **numpy.random.shuffle()** para embaralhar a sequência de 0 a 9. A cada iteração do loop interno (**stimuli**), usamos o método **list.pop()** para selecionar um valor da lista e apresentar na tela ao participante. Como a lista não é atualizada a cada tentativa, trata-se de uma espécie de amostragem **sem** reposição (na versão anterior, tínhamos uma amostragem **com** reposição). Isso garante que algarismos não se repetirão em uma mesma sequência da tarefa;
* Exemplo 3—backward digit span:
  + Usa o fatiamento de strings para inverter a ordem dos dígitos que serão comparados com a resposta do participante (i.e., **string[::-1]**):

- revisão: Conceitos **novos** e **revisitados** na aula de hoje:

* **Uso de listas para armazenar strings com as instruções, de contador (que monitora em qual tela da instrução estamos) e do** **visual.ButtonStim** **para apresentar botões com frescurinhas na tela;**
* **Uso de loops aninhados em três níveis, para controlar os dígitos apresentados em cada tentativa (stimuli), o número de tentativas com um mesmo comprimento de dígitos (span\_trials) e as tentativas com diferentes comprimentos de dígitos do digit span task (trials);**
* **Uso dos comandos numpy.random.randint() e do numpy.random.shuffle() para sortear um valor inteiro dentro de dado intervalo e para embaralhar os valores de uma lista, respectivamente. Tais comandos são úteis para gerenciar os dígitos que serão apresentados no digit span task;**
* **Uso do comando list.pop() para remover um elemento de uma lista embaralhada e atribuí-lo a uma string que guardará a sequência apresentada ao participante;**
* **Controle do número (variável) de repetições de um loop por meio do uso de uma variável do Excel (nReps = series\_len);**
* **Uso de uma lista (allowed\_keys) para indicar quais teclas são permitidas de serem usadas como resposta no teclado;**
* **Uso de código para controlar a entrada de textos no teclado e sua respectiva apresentação na tela;**
* **Uso de thisExp.addData() para salvar determinadas variáveis importantes no arquivo de saídas do programa;**
* **Checagem, após cada duas tentativas de um mesmo comprimento de dígitos, se o participante cumpriu o critério para avançar ou não para a próxima tentativa. Tal controle foi feito por meio do uso do atributo loop.thisN (span\_trials.thisN == 1);**
* **Encerramento de um loop por meio de uma estrutua condicional if e do comando loop.finished = True;**
* **Uso do fatiamento de strings (i.e., string[::-1]) para inverter a ordem dos dígitos (no backward digit span), de modo a comparar a sequência na ordem inversa de apresentação com a resposta do participante na ordem respondida.**