TEST DEL QI

Ripamonti Daniele, Serughetti Denis, Bonaffini Manuel, Garish Mohamed
3Ai - A.S. 24/25

INDICE

TESTO DEL PROBLEMA

Si vuole realizzare un programma che calcoli il quoziente intellettivo: dopo aver risposto a 20 domande si ha il risultato del proprio test, che viene messo a confronto con la media degli altri.

OVERWIEW DELLA SOLUZIONE

Per realizzare questa app viene utilizzato python con la libreria flet e Visual studio code come editor. Per la documentazione abbiamo scelto di usare canva .
Per implementare dei nuovi elementi di flet è stato fatto uso della documentazione flet e chatGPT.
Le immagini del programma sono state prese dal test proposto dal sito mensa.it

DOCUMENTAZIONE TECNICA

Per realizzare l'app abbiamo usato la libreria flet, la sleep della libreria random e i thread della libreria threading.
L'applicazione è divisa in tre parti:

- Nella prima si trovano tutte le funzioni
- Nella seconda vengono inizializzate le variabili e gli elementi di flet
- Nella terza vengono messi a schermo i vari elementi

FUNZIONE MAIN

La funzione main è essenziale per ogni programma realizzato con flet in quanto permette di creare la finestra dell'applicazione.
Dentro questa funzione si trova tutto il programma, incluse le altre funzioni.
Nell'immagine è mostrata solo la parte iniziale della funzione, che va invece avanti per più di 300 righe.

```python
def main(page: ft.Page):
    #impostazioni base della pagina
    page.window.width = 1200
    page.window.height = 900
    page.title = "Test del quoziente intellettivo"
    page.vertical_alignment = ft.MainAxisAlignment.START

    def carica_domanda(n_domanda): #per ogni domanda prende immagini e risposte
        avanti = False
        try:
            f = open("percorsi_foto.txt", "r", encoding="utf-8")
            avanti = True
        except FileNotFoundError:
            print("file non trovato")

        if avanti:
            percorsi = []
            x = 0
            # con questo ciclo si contano i pipe (|) per trovare la posizione dei percorsi
            for r in f:
                r = r.strip("\n")
                if r == "|":
                    x += 1
                if x == n_domanda:
                    for i in range(8):
                        percorsi.append(f.readline().strip("\n"))
                    break

            f.close()
            return percorsi

    def carica_dati(range_eta): #prende i dati dei test qi già svolti
        errore = True
        try:
            f = open("risultati.txt", "r", encoding="utf-8")
            errore = False
        except:
            print("errore: file non trovato")
```

FUNZIONE CARICA_DOMANDA

Questa funzione prende in ingresso la domanda alla quale ci si trova per restituire i percorsi delle immagini della domanda e la risposta corretta. Per fare ciò viene usato un file percorsi_foto.txt dove si trovano i vari percorsi

```python
def carica_domanda(n_domanda): #per ogni domanda prende immagini e risposte
    avanti = False
    try:
        f = open("percorsi_foto.txt", "r", encoding="utf-8")
        avanti = True
    except FileNotFoundError:
        print("file non trovato")

    if avanti:
        percorsi = []
        x = 0
        # con questo ciclo si contano i pipe (|) per trovare la posizione dei percorsi
        for r in f:
            r = r.strip("\n")
            if r == "|":
                x += 1
            if x == n_domanda:
                for i in range(8):
                    percorsi.append(f.readline().strip("\n"))
                break

        f.close()
        return percorsi
```

FUNZIONE CARICA_DATI

Questa funzione prende in ingresso un range d'età e restituisce il quoziente intellettivo medio generale e quello medio per il range d'età specificato. Per fare ciò, legge i dati dal file risultati.txt, dove ogni riga contiene un punteggio QI e il relativo range d'età separati da "|". I punteggi vengono salvati in una lista, e quelli appartenenti al range indicato vengono salvati in una seconda lista. Alla fine, la funzione calcola la media di entrambe le liste e le restituisce.

```python
def carica_dati(range_eta): #prende i dati dei test qi già svolti
    errore = True
    try:
        f = open("risultati.txt", "r", encoding="utf-8")
        errore = False
    except:
        print("errore: file non trovato")

    if not(errore):
        tot = []
        tot_eta = []

        #questo ciclo aggiunge alla lista tot tutti i risulati e a
        for r in f:
            r = r.replace("\n", "")
            dati = r.split("|")
            tot.append(int(dati[0]))
            if dati[1] == range_eta:
                tot_eta.append(int(dati[0]))
        f.close()

        # calcolo qi medio generale e nel range d'età
        somma = 0
        for i in tot:
            somma += i
        media_tot = somma // len(tot)

        somma = 0
        for i in tot_eta:
            somma += i
        media_eta = somma // len(tot_eta)

        return media_tot, media_eta
```

FUNZIONE SALVA_DATI

Questa funzione prende in ingresso il punteggio ottenuto nel test e il range d'età del partecipante, e li salva all'interno del file risultati.txt aprendolo in modalità aggiunta.

```
def salva_dati(punteggio, range_eta): # aggiunge sul file i
    errore = True
    try:
        f = open("risultati.txt", "a", encoding="utf-8")
        errore = False
    except:
        print("errore: file non trovato")

    if not(errore):
        f.write(str(punteggio) + "|" + range_eta + "\n")
        f.close()
```

FUNZIONE CREA_DOMANDA

Questa funzione prende in ingresso la domanda alla quale ci si trova e, servendosi della funzione carica_domanda, applica i vari percorsi alle immagini e tiene conto della risposta corretta.

```
def crea_domanda(n_domanda): # prende le domar
    percorsi = carica_domanda(n_domanda)
    img_domanda.src = percorsi[0]

    for i in range(len(img_opzioni)):
        img_opzioni[i].visible = True
        img_opzioni[i].src = percorsi[i + 1]

    risposte_corrette.append(percorsi[7]) # sa
```

FUNZIONE ATTIVA_INIZIO

Questa funzione viene eseguita quando si preme il pulsante per iniziare il test e mostra il layout del test. Inoltre, avvia il timer e imposta il punteggio iniziale a zero.

```
def attiva_inizio(e): # una volta
    if e.data != None:
        inizio.disabled = False
        inizio.update()
```

FUNZIONE START

Questa funzione si occupa di avviare il test. Una volta chiamata, rimuove gli elementi iniziali dell'interfaccia (come il titolo, il menu a tendina e il pulsante di avvio) e mostra gli elementi necessari per lo svolgimento del test.

```
def start(e): # rimuove gli elementi mostra
    titolo1.visible = False
    testo1.visible = False
    testo2.visible = False
    inizio.visible = False
    range_età.visible = False
    testo_domanda.visible = True
    domanda.visible = True
    img_domanda.visible = True
    opzioni.visible = True
    successivo.visible = True
    precedente.visible = True
    termina.visible = True
    timer.visible = True
    crea_domanda(domanda.value) # prende i
    for i in range(len(img_opzioni)):
        img_opzioni[i].visible = True
    crea_thread(e)
    page.update()
```

FUNZIONE INIZIO_TIMER

Questa funzione imposta e avvia un timer di 15 minuti per il test. Il timer viene aggiornato ogni secondo e mostrato sull'interfaccia utente. Quando il timer raggiunge lo zero, il test termina automaticamente e viene richiamata la funzione per concludere il test.

```
def inizio_timer(e): # fa iniziare il timer
    print()
    global timer_attivo, secondi, minuti
    timer_attivo = True
    # il ciclo serve per far scorrere il tempo
    while timer_attivo:
        secondi -= 1
        if minuti == 0 and secondi == -1:
            secondi = 0
            timer_attivo = False
        elif secondi == -1:
            secondi = 59
            minuti -= 1

        timer.value = f"- Tempo rimasto {minuti} : {secondi}"
        page.update()
        sleep(1)

    # una volta scaduto il timer si il test finisce
    for i in range(len(risposte_corrette), 20): # serve a far
        risposte_corrette.append(None)

    risultati(e)
    fine(e)
```

FUNZIONE CREA_THREAD

Questa funzione crea un thread che esegue il conteggio del tempo tramite la funzione inizio_timer, permettendo al programma di continuare ad eseguire altre operazioni senza interruzioni.

```
def crea_thread(e): # crea un thread
    global timer_attivo
    thread_timer = Thread(target=inizio_timer(e), daemon=True)
    thread_timer.start()
    timer_attivo = True
```

FUNZIONE SUCC

Questa funzione viene chiamata quando si preme il pulsante “successivo”. permette di passare alla domanda successiva: salva in risposte utente l’opzione scelta e poi usa la crea_domanda per passare alla domanda successiva. se la domanda è la 20 si disabilita il pulsante “successivo”, se è maggiore di 1 attiva il pulsante “precedente”.

```
def succ(e): # passa alla domanda successiva salvando la rispo
    if opzioni.value == None:
        opzioni.value = ""
    if opzioni.value != "":
        risposte_utente[domanda.value - 1] = opzioni.value

    domanda.value += 1
    opzioni.value = risposte_utente[domanda.value - 1]
    crea_domanda(domanda.value)

    # se si trova alla domanda 20 disabilita il pulsante succe
    if domanda.value == 20:
        successivo.disabled = True
    if domanda.value > 1:
        precedente.disabled = False
    page.update()
```

FUNZIONE PREC

Questa funzione viene chiamata quando si preme il pulsante “precedente”. Svolge la stessa funzione di succ, solo che invece di passare alla domanda successiva va a quella precedente. Se la domanda è la 1 si disabilita il pulsante “precedente”, se è minore di 20 attiva il pulsante “successivo”.

```
def prec(e): # passa alla domanda precedente salvando la ris
    if opzioni.value == None:
        opzioni.value = ""
    if opzioni.value != "":
        risposte_utente[domanda.value - 1] = opzioni.value

    domanda.value -= 1
    opzioni.value = risposte_utente[domanda.value - 1]
    crea_domanda(domanda.value)

    # se si trova alla domanda 1 disabilita il pulsante prec
    if domanda.value == 1:
        precedente.disabled = True
    if domanda.value < 20:
        successivo.disabled = False
    page.update()
```

FUNZIONE CONTROLLA_TERMINA

Questa funzione viene chiamata quando si preme il tasto termina: dopo aver salvato la risposta della domanda corrente, si verifica se si ha risposto a tutte le domande. In tal caso si passa alla messa a schermo dei risultati, altrimenti viene aperto un popup che indica che non si ha risposto a tutte le domande.

```
def controlla_termina(e): # verifica che l'utente abbia risp
    if opzioni.value == None:
        opzioni.value = ""
    if opzioni.value != "":
        risposte_utente[domanda.value - 1] = opzioni.value

    if "" in risposte_utente: # se ci sono delle domande sen
        apri_popup(e)

    else:
        risultati(e)
        fine(e)

    page.update()
```

FUNZIONE APRI_POPUP

Questa funzione viene chiamata quando si preme il tasto termina ma non si ha risposto a tutte le domande.
Viene aperto un popup che indica quali sono le domande senza risposta.

```
def apri_popup(e): # funzione che apre il popup
    # con questo ciclo vengono trovate le domand
    testo = "Non hai risposto alle domande: "
    for i in range(len(risposte_utente)):
        if risposte_utente[i] == "":
            testo += str(i + 1) + ", "
    testo = testo[:-2]
    popup.content = ft.Text(testo, size = 16)

    page.open(popup)
    page.update()
```

FUNZIONE RISULTATI

Questa funzione confronta le risposte fornite con quelle corrette e calcola il punteggio del QI. Viene poi presa la media generale insieme a quella nel proprio range d'età, in modo da avere dei risultati di riferimento.
Se la media è maggiore del punteggio viene colorata di rosso, altrimenti viene colorata di verde

```python
def risultati(e): # calcola il qi in base alle riosposte e prende i dati del file per
    # calcolo punteggio
    punteggio = 0
    for i in range(len(risposte_utente)):
        if risposte_utente[i] == risposte_corrette[i]:
            punteggio += 3

    # i minori di 16 anni e le persone nel range 16 - 20 hanno un piccolo aumento del
    if range_età.value == "< 16":
        punteggio += punteggio * 0.25
    elif range_età.value == "16 - 20":
        punteggio += punteggio * 0.15
    punteggio = int(round(punteggio, 0))
    punteggio += 70
    qi.value = f"Il tuo QI è {punteggio}"

    #prende i risultati medi
    media_tot, media_eta = carica_dati(range_età.value)

    qi_medio.value = f"QI medio: {media_tot}"
    qi_medio_eta.value = f"QI medio nel range d'età {range_età.value} : {media_eta}"

    if media_tot > punteggio:
        qi_medio.color = "red"
    else:
        qi_medio.color = "green"
    if media_eta > punteggio:
        qi_medio_eta.color = "red"
    else:
        qi_medio_eta.color = "green"

    # salva il punteggio dopo aver calcolato la media (si esclude così il proprio pun
    salva_dati(punteggio, range_età.value)
```

FUNZIONE FINE

Questa funzione nasconde gli elementi del test e mostra i risultati finali, inclusi il punteggio del QI e le medie, per concludere il test e fornire un riepilogo all'utente.

```python
def fine(e): # rimuove gli elementi del test per mostrare i c
    testo_domanda.visible = False
    domanda.visible = False
    img_domanda.visible = False
    opzioni.visible = False
    successivo.visible = False
    precedente.visible = False
    termina.visible = False
    riepilogoText.visible = True
    riepilogoText.value = "Test completato con successo!"
    qi.visible = True
    qi_medio.visible = True
    qi_medio_eta.visible = True
    timer.visible = False
    for i in range(len(img_opzioni)):
        img_opzioni[i].visible = False
    page.vertical_alignment = ft.MainAxisAlignment.CENTER
    page.update()
```

TEST ESEGUITI

Sono stati eseguiti vari test per capire se ci fosse qualche bug o qualche errore:

- Non vengono mai lanciate eccezioni
- Domande, immagini e risposte non sono sfalsate
- I calcoli di punteggio e medie sono corretti
- Non ci sono problemi con caricamento e salvataggio dei dati
- Il timer funziona correttamente e al suo scadere termina la prova

FOCUS SU CHI HA FATTO COSA

Ripamonti: Ha realizzato il corpo principale del programma, ha gestito la parte di immagini e file, ha realizzato metà documentazione

Serughetti: Ha realizzato diverse funzioni e ha realizzato l'altra metà di documentazione

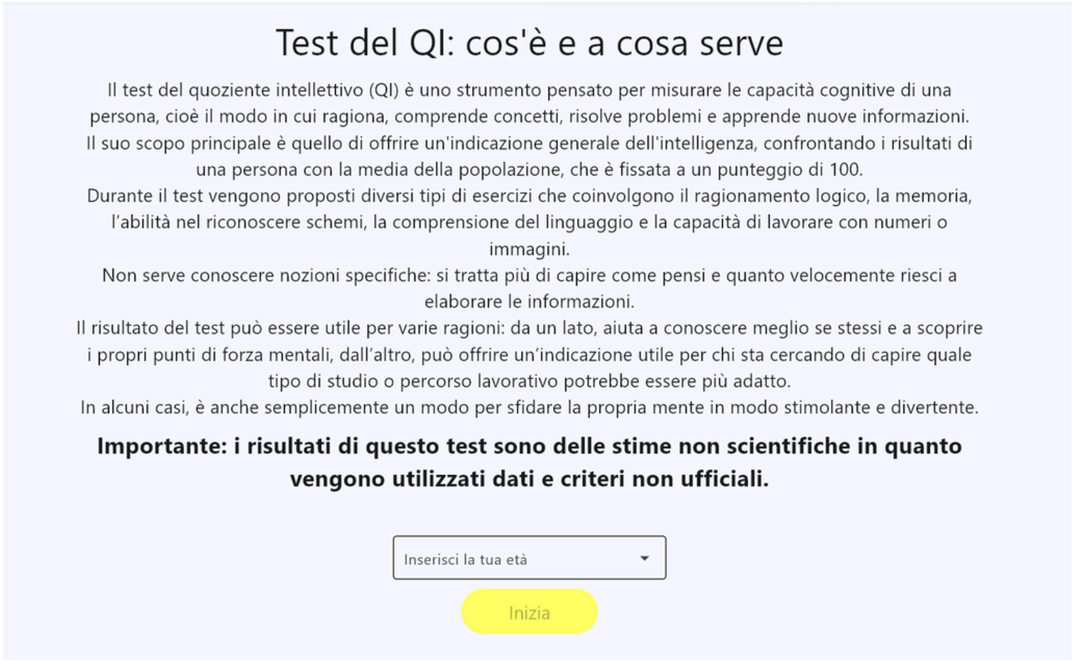
Bonaffini: Non ha contribuito

Garish: Ha guardato dei video di base su python svolgendo gli esercizi che gli venivano proposti

DOCUMENTAZIONE UTENTE

DOCUMENTAZIONE UTENTE

Questa applicazione permette di calcolare il proprio quoziente intellettivo (QI) attraverso un test composto da 20 domande.
All'avvio del test, viene mostrata una breve introduzione su cos'è il QI e viene richiesto all'utente di inserire la propria età.
Inserita l'eta si potrà andare a rispondere alla prima domanda.



DOCUMENTAZIONE UTENTE

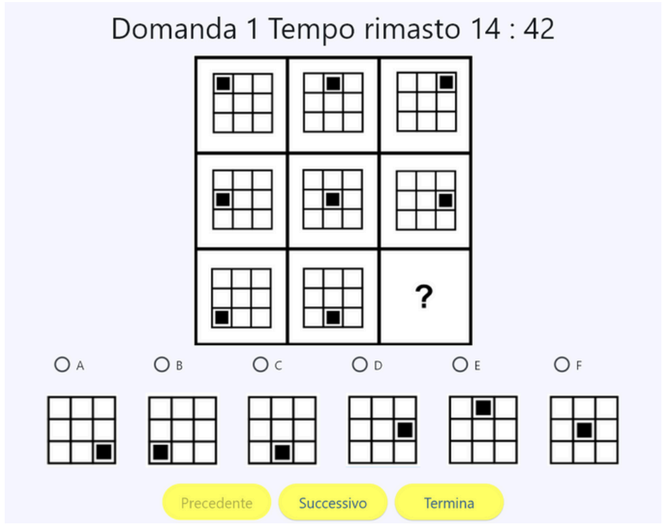
Il test si compone di 20 domande, ognuna con una sola risposta possibile.
L'utente può navigare tra le domande usando tre pulsanti.

-Precedente: torna alla domanda precedente

-Successivo: passa alla domanda successiva

-Termina: consente di concludere il test in qualsiasi momento

È presente anche un timer visibile, che indica quanto tempo resta per completare il test. Questo aiuta l'utente a gestire meglio le proprie risposte.



DOCUMENTAZIONE UTENTE

Se l'utente preme "Termina" senza aver risposto a tutte le domande, compare un pop-up che segnala esattamente quali domande sono rimaste in bianco. Se tutte le risposte sono state fornite, l'app mostra il risultato del QI calcolato in base alle risposte date, insieme a un confronto con la media degli altri utenti.