

# Narzędzia Pracy Grupowej

## Sprawozdanie z projektu

### *Temat projektu i funkcjonalność*

Tematem naszego projektu był „Blackjack”, w którym musieliśmy spełnić poniższe punkty:

- 1) Implementacja gry do jednej talii do 8.
- 2) Gra z komputerem.
- 3) Statystyki wygranych zapamiętywanych nawet po wyłączeniu programu.
- 4) Możliwość cofnięcia ruchu.
- 5) Tryb gry na czas.
- 6) Jedna zaproponowane przez grupę: otrzymanie przez gracza odpowiedniej rangi, zależnej od procentowych wygranych.

Blackjack to kasynowa gra karciana, w której gracz stara się pokonać krupiera poprzez uzyskanie sumy jak najbliższej 21 punktów w kartach jednak nie przekraczając 21.

### *Skład grupy*

-Mateusz Szynal  
-Szymon Tokarz  
-Filip Stefaniak  
-Aleksandra Ślęzak

### *Zasady gry*

Blackjack jest kasynową wersją gry w oczko. Zadaniem gracza jest uzyskać jak najbliższej (ale nie więcej niż) 21 punktów. Najwyższym układem kart jest tzw. Blackjack, czyli as i 10 lub figura, za który gracz dostaje 150% zakładu. W grze używa się kilku talii złożonych z 52 kart. Używa się ich od jednej aż do ośmiu. Regułą jest, że im mniej tym lepiej dla gracza.

### *Wstęp*

Do rozdzielania zadań skorzystaliśmy z Asany, gdzie Mateusz stworzył zespół i dodał resztę członków. Po skonsultowaniu się z grupą rozdzielił on poszczególne zadania projektu do wykonania w określonym czasie. Asana umożliwiła nam sprawną wizualizację projektu jako lista, kalendarz, tablica oraz pogrupowanie zadań. W realizacji naszego zadania niezawodny był GitHub, którego wykorzystaliśmy do zarządzania kodem oraz rozproszonej kontroli wersji. Zapewnił on nam kontrolę dostępu oraz kilka funkcji współpracy, takich jak śledzenie błędów, zarządzanie zadaniami w naszym wspólnym repozytorium. Cały kod wykonaliśmy w języku Python, który był ogólnie zalecany przez prowadzącego.

Podczas wykonywania zleconego projektu pomijając komunikację zdalną przez różne kanały komunikacji m.in.: Messenger, Discord, odbywaliśmy częste spotkania z całym zespołem głównie ze względu na bliskie zamieszkanie (w tym samym akademiku). Nasza współpraca była bezproblemowa, ponieważ wszyscy dążyliśmy do sprawnego i satysfakcjonującego wykonania wyznaczonego celu.

***Role metoryczne:***

- Kierownik zespołu - Mateusz Szynal
- Kierownik projektu – Szymon Tokarz
- Programista – Filip Stefaniak
- Reprezentant/osoba odpowiedzialna za dokumentację – Aleksandra Ślęzak

***Role nieformalne w ujęciu Belbina:***

Każdy z członków zespołu ma coś w sobie z każdego typu przedstawicieli w ujęciu Belbina natomiast główne cechy zdecydowały, że:

- Realizator – Szymon Tokarz, który wykazał się sumiennością, cierpliwością we wspieraniu innych członków zespołu w trudnych sytuacjach, a także skupił się na wyznaczonym celu i dążył do jego spełnienia

Zalety: umiejętność przystosowania się, zorganizowanie, praktyczność

- Koordynator – Mateusz Szynal, który kierował pracą całego zespołu i rozwiązywał na bieżąco występujące problemy

Zalety: dążenie do celu, ufność, opanowanie

- Krytyk wartościujący - Filip Stefaniak, który podejmował optymalne decyzje, prowadził szczegółowe analizy problemów i celów, jego działanie przyczyniło się do wyeliminowania błędów w projekcie

Zalety: pragmatyczność, opanowanie

- Dusza zespołu - Aleksandra Ślęzak, która zapobiegała konfliktom w zespole, wspierała wszystkich członków oraz dzielnie napędzała do pracy

Zalety: entuzjazm, komunikatywność

Niestety cały projekt rozłożyliśmy w czasie ze względu na czynniki od nas niezależne. Autorami funkcji poniżej jest praktycznie każdy z nas, ponieważ ilość wersji które pojawiły się na naszym gicie, jest dość spora. Często zmienialiśmy koncepcje wyglądu całości. Poniższy autorzy stworzyli zarys działania konkretnej funkcji.

### ***1. Ustalenie początkowych informacji potrzebnych do rozpoczęcia gry:***

Wykonał: Filip Stefaniak

Po zrobieniu research'u w Google, rozpoczęliśmy pracę z programem. Pierwszym krokiem było wykonanie funkcji, z której będzie wiadoma ilość używanych talii kart do gry, a także inicjalizacja wygranych oraz przegranych podczas jednego ciągu gier.

```
# import os
import random
import time
decks=0
while int(decks)<1 or int(decks)>8:
    decks = input("Wybierz liczbę talii:\t")
    if int(decks)<1 or int(decks)>8:
        print("Zła liczba talii, proszę wybierz odpowiednią liczbę talii")
# użycie chcianej liczby talii

deck = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14] * (int(decks) * 4)

# inicjalizacja wygranych oraz przegranych podczas jednego ciągu gier

wins = 0
losses = 0
```

## ***2. Przywitanie z graczem***

Wykonał: Szymon Tokarz oraz Mateusz Szynal

Z racji dobrego wychowania, stwierdziliśmy, że doskonałym pomysłem będzie wykonanie funkcji, która przywita gracza, a także przekaże mu istotne informacje co do dalszego postępowania w rozgrywce.

```

# przywitanie sie z graczem
def hello():
    print("\n\t\t\t\t Witaj w Blackjack!\n")
    menu = input("[G]ra, [S]tatystyki, [Z]asady, [R]anga, [W]yjdz:\t")
    if menu == "g":
        tryb = input("Wybierz tryb gry: [c]zas/[b]ez czasu:\t")
        # if tryb== "c":
        # tryb gry na czas
        if tryb == "b":
            game()
        else:
            hello()
    elif menu == "s":
        print("\tWYGRANE W RUNNIE: " + str(wins) + "\tPRZEGRANE W RUNNIE: " + str(losses) + "\n")
        file = open("output.txt", "r")
        win = 0
        all_ = 0
        for x in file:
            if int(x) == 1:
                win = win + 1
            all_ = all_ + 1
        print("\tWYGRANE OGOLNIE: " + str(win) + "\tPRZEGRANE OGOLNIE: " + str(all_ - win) + "\n")
        quit = input("Kliknij q, aby wrócić do menu\t" + "\n")
        if quit == "q":
            hello()
    elif menu == "z":
        print("https://pl.wikipedia.org/wiki/Blackjack")
        quit = input("Kliknij q, aby wrócić do menu\t")
        if quit == "q":
            hello()
    elif menu == "r":
        rank()
        quit = input("Kliknij q, aby wrócić do menu\t")
        if quit == "q":
            hello()
    elif menu == "w":
        print("Do zobaczenia")
        exit()
    else:
        hello()

```

### 3. Funkcja określająca stopień zaawansowania gracza

Wykonał: Filip Stefaniak

Dodatkiem wymyślonym przez nas zespół była funkcja określająca stopień zaawansowania gracza. Określa ona jego poziom po 20 grach według których jest później przyznawana dana ranga.

```

def rank():
    file = open("output.txt", "r")
    win = 0
    all_ = 0
    for x in file:
        if int(x) == 1:
            win = win + 1
            all_ = all_ + 1
    percentage = win / all_ * 100
    print("\tWYGRANE OGOLNIE: " + str(win) + "\tPRZEGRANE OGOLNIE: " + str(all_ - win) + "\tPROCENT OGOLNIE: " + str(
        percentage) + "%\n")
    if all_ < 20:
        print("potrzebujesz 20 gier aby mieć range")
        return 0
    if percentage < 20:
        print("Brąz")
    elif 20 <= percentage < 40:
        print("Srebro")
    elif 40 <= percentage < 60:
        print("Złoto")
    elif 60 <= percentage < 80:
        print("Płatyna")
    elif 80 <= percentage < 100:
        print("Diament")
    elif percentage == 100:
        print("Mistrz")

```

#### ***4. Funkcja przypisująca dwie początkowe karty oraz funkcja sumująca wartość danego posiadacza kart***

Wykonał: Filip Stefaniak

Funkcje te są bazą do funkcjonowania tej gry. W obu przypadkach użyliśmy pętli for, która ułatwiła, a także skróciła zapis kodu. Dla przejrzystości, w odpowiednich

momentach funkcji dodaliśmy po “#” komentarze, które umożliwiły nam szybkie i sprawne poruszanie się po naszej pracy.

```
def deal(deck, number):
    hand = []
    for i in range(number):
        random.shuffle(deck)
        card = deck.pop()
        if card == 11:
            card = "J"
        if card == 12:
            card = "Q"
        if card == 13:
            card = "K"
        if card == 14:
            card = "A"
        hand.append(card)
    return hand
```

```
# funkcja ktora sumuje wartosc danego posiadacza kart
def total(hand):
    total = 0
    # J, Q i K maja wartosc rowna 10
    for card in hand:
        if card == "J" or card == "Q" or card == "K":
            total += 10
            # wartosc asa jest zalezna od tego ile wart jest deck
        elif card == "A":
            total = total + 11
            if total > 21:
                total = total - 10
        # jesli zadne z powyzzszych to po prostu dodawana jest wartosc karty
        else:
            total += card
    return total
```

***5. Dodanie kolejnej karty podczas HIT, funkcja cofania ruchu, funkcja wpisująca statystyki, bez wyniku rezultatu oraz funkcja wpisująca na samym końcu szczegóły końcowego wyniku gry***

Wykonał: Filip Stefaniak, Szymon Tokarz i Mateusz Szynal

Kolejne szczegóły bez których gra nie miałyby sensu.

```
# dodanie jednej kolejnej karty podczas HIT
def hit(hand):
    card = deck.pop()
    if card == 11:
        card = "J"
    if card == 12:
        card = "Q"
    if card == 13:
        card = "K"
    if card == 14:
        card = "A"
    hand.append(card)
    return hand
```

```
# cofanie ruchu
def back(hand):
    back = input("Czy chciałbyś cofnąć ruch? [t]/[n]\t")
    if back == "t":
        del hand[-1]
    return hand
```

```
def print_results(dealer_hand, player_hand):
    # print("\tWINS: " + str(wins) + "\tLosses: " + str(losses)+ "\n")
    print("\n")
    print("Dealer punkty:\t" + str(total(dealer_hand)) + "\t i posiada karty:\t" + str(dealer_hand))
    print("Twoje punkty:\t" + str(total(player_hand)) + "\t i posiada karty:\t" + str(player_hand))
```

## ***6. Funkcja odpowiedzialna za wpisanie końcowego rezultatu rozgrywki***

*Wykonał Szymon Tokarz*



```
def score(dealer_hand, player_hand):  
    global wins  
    global losses  
  
    # najpierw sprawdzany jest warunek czy gracz nie ma równo 21 punktów  
  
    if total(player_hand) == 21:  
        print_results(dealer_hand, player_hand)  
        print("Gratulacje! Masz Blackjacka!\n")  
        wins += 1  
        file = open("output.txt", "a")  
        file.write("1\n")  
        file.close()  
  
    # następnie sprawdzany jest warunek czy dealer ma 21 pkt  
    elif total(dealer_hand) == 21:  
        print_results(dealer_hand, player_hand)  
        print("Niestety, dealer ma blackjacka. Przegrales.\n")  
        losses += 1  
        file = open("output.txt", "a")  
        file.write("0\n")  
        file.close()  
  
    # następnie sprawdzane jest czy gracz nie ma więcej niż 21 punktów  
    elif total(player_hand) > 21:  
        print_results(dealer_hand, player_hand)  
        print("Niestety, masz więcej niż 21 punktów. Przegrales.\n")  
        losses += 1  
        file = open("output.txt", "a")  
        file.write("0\n")  
        file.close()  
  
    # sprawdzenie czy dealer ma więcej niż 21 punktów  
    elif total(dealer_hand) > 21:  
        print_results(dealer_hand, player_hand)  
        print("Dealer ma więcej niż 21 punktów. Wygrałeś!\n")  
        wins += 1  
        file = open("output.txt", "a")  
        file.write("1\n")  
        file.close()
```

```
# sprawdzenie czy gracz ma mniej punktow niz dealer
elif total(player_hand) < total(dealer_hand):
    print_results(dealer_hand, player_hand)
    print("Niestety. Twój wynik jest niższy niż dealera. Przegrales.\n")
    losses += 1
    file = open("output.txt", "a")
    file.write("0\n")
    file.close()

# sprawdzenie czy gracz ma więcej punktow niz dealer
elif total(player_hand) > total(dealer_hand):
    print_results(dealer_hand, player_hand)
    print("Gratulacje. Masz wyższy wynik niż dealer. Wygrałeś!\n")
    wins += 1
    file = open("output.txt", "a")
    file.write("1\n")
    file.close()
elif total(player_hand) == total(dealer_hand):
    print_results(dealer_hand, player_hand)
    print("REMIS\n")
```

## 7. Funkcja odpowiedzialna za grę:

Wykonał: Szymon Tokarz, Mateusz Szynal

Funkcja łącząca funkcjonalności poprzednich funkcji.

```
def game():
    global wins
    global losses
    choice = 0

    dealer_hand = deal(deck, 2)
    player_hand = deal(deck, 2)

    print("\nDealer pokazuje " + str(dealer_hand[0]))

    print("Masz następujące karty " + str(player_hand) + " o liczbie punktów: " + str(total(player_hand)) + "\n")

    if total(player_hand) == 21:
        score(dealer_hand, player_hand)
        back_to_menu()

    quit = False

    while not quit:
        choice = input("Do wyboru: [H]it, [S]tand, or [Q]uit: \t").lower()
        if choice == 'h':
            hit(player_hand)
            print("Twoje karty: " + str(player_hand))
            print("Twoje punkty: " + str(total(player_hand)) + "\n")
            back(player_hand)
            if total(player_hand) > 21:
                if "A" in player_hand:
                    sum = total(player_hand) - 10
                    if sum > 21:
                        score(dealer_hand, player_hand)
                        back_to_menu()
                    else:
                        score(dealer_hand, player_hand)
                        back_to_menu()
                else:
                    score(dealer_hand, player_hand)
                    back_to_menu()
            elif total(player_hand) == 21:
                score(dealer_hand, player_hand)
                back_to_menu()
```

```

elif choice == 's':
    while total(dealer_hand) < 17:
        print("Dealer dobiera karte")
        hit(dealer_hand)
        if total(dealer_hand) > 21:
            score(dealer_hand, player_hand)
            back_to_menu()
        score(dealer_hand, player_hand)
        back_to_menu()
elif choice == "q":
    print("Do zobaczenia!")
    exit()

```

### 9. Funkcja game ale z obsługą czasu rozgrywki:

Wykonał: Filip Stefaniak

W tym przypadku osoba grająca ma 10 sekund na podjęcie decyzji o tym co chce zrobić.

```

def game_ale_z_czasem():
    global wins
    global losses
    t=time.time()
    choice = 0

    dealer_hand = deal(deck, 2)
    player_hand = deal(deck, 2)
    int

    print("\nDealer pokazuje " + str(dealer_hand[0]))

    print("Masz następujące karty " + str(player_hand) + " o liczbie punktów: " + str(total(player_hand)) + "\n")

    if total(player_hand) == 21:
        score(dealer_hand, player_hand)
        back_to_menu()

    quit = False

    while not quit:
        t=time.time()
        choice = input("Do wyboru: [H]it, [S]tand, or [Q]uit: \t").lower()
        if time.time()-t>=10:

            print("Twój czas dobiegł końca")
            losses += 1
            file = open("output.txt", "a")
            file.write("0\n")
            file.close()
            back_to_menu_ale_z_czasem()

```

## ***8. Funkcja odpowiedzialna za powrót do menu:***

Wykonał: Szymon Tokarz

Funkcja pozwalająca rozpocząć nową rozgrywkę.

```
# powrot do menu
def back_to_menu():
    back = input("Czy chcesz wrócić do menu [t]/[n]\t").lower()
    if back == "t":
        hello()
    else:
        game()
```

## ***Wnioski***

Wykonanie tego projektu nauczyło nas jak pogodzić różne charaktery podczas pracy w zespole, rozwiązywać powstałe konflikty oraz rozwiązywać problemy, a także merge'wać pracę. Dużym ułatwieniem do komunikacji okazał się GitHub, a także Messenger. Każdy z członków zespołu wykonał dobrze powierzoną mu część projektu.

Dokumentację wykonała

Aleksandra Ślęzak