

Dawid Bitner MS INF 1A

HANGMAN [C]

Dokumentacja projektu

Spis treści

Część I	2
Opis projektu	2
Środowisko testowe.....	2
IDE i kompilator.....	2
Instrukcja obsługi	2
Singleplayer.....	3
Multiplayer.....	4
Część II	5
Specyfikacja techniczna	5
Podział projektu na funkcje	5
Zmienne globalne	5
Biblioteki.....	5
Szczegóły techniczne	5
Działanie programu.....	5
Szkielet gry, algorytm funkcji gra();	6

Część I

Opis projektu

HANGMAN[C] jest prostą grą konsolową napisaną w języku C. Program umożliwia prowadzenie rozgrywki zarówno w trybie Single jak i Multiplayer. Tryb Singleplayer posiada zaimplementowane 3 poziomy trudności, na każdy poziom przypada 15 różnych słów do odgadnięcia, każde słowo posiada wskazówkę która naprowadza nas na właściwą myśl.

Tryb Multiplayer polega na tym, że pierw wprowadzamy hasło, wskazówkę i wybieramy liczbę szans 5, 10 lub 15, następnie ekran jest czyszczony i druga osoba może odgadywać hasło.

Środowisko testowe

Program działa na systemie Windows Vista, lub nowszym. Został przetestowany w dwóch środowiskach:

- Windows 7 Ultimate 64bit
- Windows 10 Home 65bit

IDE i kompilator

IDE: **Dev-C++ v 5.11**

kompilator: **TDM-GCC 4.9.2 64-bit**

Instrukcja obsługi

Program możemy uruchomić bezpośrednio poprzez wybranie pliku **wisielec.exe**, lub z poziomu konsoli wpisując jego lokalizację: [LOKALIZACJA]\wisielec.exe.

Uwaga: zaleca się granie w trybie pełnoekranowym, poprzez kliknięcie ikony powiększenia konsoli cmd zostanie ona rozciągnięta do 100% wysokości ekranu, jest to zalecane, gdyż w ten sposób mamy pewność że cała gra zmieści nam się na ekranie i nie będziemy zmuszeni do używania paska przewijania.

Po uruchomieniu programu ukazuje nam powitalny ASCII-ART, wraz z zapytaniem o tryb rozgrywki.

```
888
888
888
88888b. 8888b. 88888b. .d88b. 88888b.d88b. 8888b. 88888b.
888 88b 88b888 88bd88P 88b888 888 88b 88b888 88b
888 888.d888888888 888888 888888 888 888.d888888888 888
888 888888 888888 888Y88b 888888 888 888888 888888 888
888 888 Y888888888 888 Y88888888 888 888 Y888888888 888
      888
      Y8b d88
      Y88P

Chcesz zeby slowo podal przeciwnik, czy zeby wygenerowal je program?
SINGLEPLAYER [S]
MULTIPLAYER  [M]
```

Ten ASCII-ART jak i inny, przedstawiający sylwetkę wisielca znajduje się pod adresem:

<http://www.ascii.co.uk/art/hangman>, autorem jest **Manus O'Donnell**

Poprzez kliknięcie klawisza **S** wybieramy tryb Singleplayer, **M** odpowiada za Multiplayer.

Singleplayer

składa się on z trzech poziomów:



Po wybraniu poziomu przechodzimy do rozgrywki. Poniżej testowany jest poziom trudny:



Odpowiednio na ekranie widzimy:

- komunikat powitalny
- wskazówkę
- ukryte, odgadywane słowo
- liczbę szans
- rysunek wisielca, który składa się z 5 etapów rysowania.

Zależnie od tego czy wygramy, czy przegramy pod rysunkiem wisielca na szubienicy pojawi się odpowiedni komunikat, oraz zostanie ujawnione odgadywane hasło:

haslo to: ANATOMOPATOLOGIA
PORAZKA!

Następnie po kliknięciu dowolnego przycisku program pyta się nas czy chcemy zagrać od nowa, czy zakończyć program

```
Gra skończona
ZAMKNIJ PROGRAM [Q]
ZAGRAJ PONOWNIE [R]
```

Multiplayer

Po wybraniu trybu mulitiplayer program pyta nas się o 3 rzeczy:

-hasło

-wskazówkę

-liczbę szans

```
MULTIPLAYER

Podaj slowo:
piłka nożna
Podaj wskazowke:
sport
Wybierz liczbe szans: [5/10/15]
10
```

Następnie ekran jest czyszczony i rozgrywka przebiega dokładnie tak samo jak po wybraniu trybu Singleplayer.

Część II

Specyfikacja techniczna

Podział projektu na funkcje

powitanie(); - odpowiada za napis powitalny

tryb_gry(); - odpowiada za wybór trybu gry

poziom_e(); - odpowiada za ilość szans i losowane słowo-zagadkę dla poziomu łatwego

poziom_m(); - j.w. dla poziomu średniego

poziom_h(); j.w. dla poziomu trudnego

gra(); - najważniejsza funkcja w programie – szkielet, odpowiada za działanie gry

rysunek_szubienicy(); - odpowiada za rysowanie wisielca za pomocą ASCII-artu

koniec_gry(); - odpowiada za pytanie gracza czy chce zakończyć rozgrywkę, czy zagrać ponownie

Zmienne globalne

Każda z funkcji posiada właściwe dla siebie zmienne, dodatkowo istnieją zmienne globalne:

char slowo[1024] – do niej wczytujemy wpisane słowo

char wskazowka[1024] – do niej wczytujemy wpisaną wskazówkę

int szanse – określa liczbę posiadanych szans

Biblioteki

W projekcie zostały użyte następujące biblioteki:

<stdio.h> - biblioteka standardowa języka C – zawiera najważniejsze funkcje

<stdlib.h> - biblioteka standardowa języka C – zawiera m.in. użytą w programie funkcję rand();

<string.h> - biblioteka odpowiadająca za operacje na łańcuchach znaków, zawiera m.in. użytą funkcję strlen(); która odpowiada za zliczanie długości wpisanego char'a.

<time.h> - zawiera funkcje obsługi czasu, użyta m.in. do wyzerowania zegara po użyciu funkcji rand(); w celu uzyskania losowego wyboru słów.

Szczegóły techniczne

Działanie programu

Działanie programu zostało opisane w instrukcji obsługi. Poniżej znajduje się wykres przechodzenia do kolejnych funkcji:

- 1) powitanie();
- 2) Tryb_gry();
 - a) poziom_e();
 - b) poziom_m();
 - c) poziom_h();
- 3) gra();
 - a) rysunek_szubienicy();
- 4) koniec_gry();

Funkcja powitanie(); występuje w kilku miejscach, m.in. po przejściu do funkcji gra(); lub koniec_gry(); odpowiada ona bowiem że to by wyczyścić ekran i za to by w jego górnej części wyświetlany był napis „HANGMAN”.

Szkielet gry, algorytm funkcji gra();

Kod programu jest dość rozległy, więc został zamieszczony tutaj tylko algorytm i opis najważniejszej funkcji w programie, odpowiadającej za jego działanie.

Kod:

```
gra(){                                //funkcja zawiera rdzeń gry;
char podlogi[1024] = {0}, litera;
int i, p, trafione_litery = 0, wynik = 0, szubienica=szanse/5, krok = 0,
rysunek = 0;

powitanie();                          //wyświetla powitanie, wraz z czyszczeniem okna, tak
by "hangman" napis w ascii-arcie był widoczny przez całą rozgrywkę;
printf("WSKAZOWKA: %s \n\n", wskazowka);    //wyświetla wskazówkę;

for(p=0; p<strlen(slowo); p++){//po wczytaniu słowa pętla tworzy bliźniaczy
char wypełniony '_' o takiej samej długości jak słowo;
slowo[p] = toupper(slowo[p]); //litery we wpisanym słowie zostają zamienione
na wielkie, po to by nie rozróżniać wielkich i małych liter;

if(slowo[p]==' '){ //jeśli w słowie występuje spacja to na początku zamiast
podłogi również pokazana jest spacja, tak by nie wprowadzać gracza w błąd
co do ilości liter;
podlogi[p]=' ';
wynik++;    //zwiększamy wynik o 1, jeżeli w słowie występuje spacja
}

else if(slowo[p]=='-'){ //jeśli w słowie występuje myślnik, to postępuj
analogicznie do sytuacji w której występuje spacja;
podlogi[p]='-';
wynik++;
}

else{    //w innym przypadku ukrywamy litery pod postacią podłóg;
podlogi[p]='_';
}
}

printf("%s",podlogi); //wypisanie na wstępie podłóg, dzięki temu gracz zna
mniej-więcej długość słowa;
printf("\n Pozostala liczba szans: %d", szanse); //wyświetla liczbę
pozostałych szans;

while(szanse){    //pętla while wykonuje się dopóki istnieją szanse;
litera=getch();
litera = toupper(litera); //wpisana litera zawsze będzie wielka, ponieważ
słowo również zostaje ustawione na same wielkie litery;

powitanie();    //czyści ekran i wyświetla hangmana;
printf("WSKAZOWKA: %s \n\n", wskazowka);    //wyświetla wskazówkę;

int temp = 0;    //zmienna tymczasowa, przy każdym wywołaniu pętli jej
wartość zostaje ustawiona na 0;

for(i=0; i<strlen(slowo); i++){ //pętla wczytuje literkę po literce;
if(litera==slowo[i]){ //jeśli litera odpowiada literze w słowie to...(1);
if(litera==podlogi[i]){ //jeśli podłoga zamieniona na literkę (2) została
podana ponownie przez użytkownika to wyświetla odpowiedni komunikat...(3);
printf("Podales litere podana wzczesniej: %c \n", litera);
temp++; //zmienna tymczasowa +1;
```

```

break; //(3) i przerywa dalszą część pętli for m.in.: po to żeby do wyniku
nie zostało dodane 1 i nie została utracona szansa.
    }
podlogi[i]=slovo[i]; //(1)(2)podlogi zamieniane są na literki;
wynik++; //wynik+1;
temp++; //zmienna tymczasowa +1;
}
}

if(!temp){ //jeśli zmienna tymczasowa nie została zwiększona to tracimy
szanse;
szanse--;
krok++; //jeśli odpowiadamy błędnie to zwiększamy krok rysowania
szubienicy;

if(krok%szubienica == 0)rysunek++; //jeśli liczba kroków jest podzielna
przez szanse/5 to zwiększamy poziom rysunku;
}

for(i=0; i<=strlen(slovo); i++){ //pętla wypisuje zamienione podlogi na
litery jeżeli zostały zamienione, jeżeli nie to podlogi;
printf("%c", podlogi[i]);
}

if(szanse>0){
printf("\n Pozostala liczba szans: %d", szanse); //wyświetla liczbę
pozostałych szans;
}

rysunek_szubienicy(rysunek); //funkcja odpowiada za rysunek wisielca na
szubienicy;

if(wynik==strlen(slovo)){ //jeśli ilość trafionych liter odpowiada długości
słowa to mamy wygraną i kończymy grę;
printf("\n%s\n WYGRALES!\n\n", slovo);
return koniec_gry(); //i przechodzimy od funkcji koniec_gry();
}

if(szanse==0){ printf("\n haslo to: %s\n PORAZKA!\n\n", slovo)//jeśli
zostaje 0 szans to wyświetlany jest komunikat o porażce, oraz szgadywane
słowo;
return koniec_gry();
}
}
printf("\n\n"); //dwa enterzy dla przyjrzyistości w grze;
}

```


Schemat blokowy

Schemat znajduje się w drugiej części dokumentacji