

GLIZDA – gra (13)

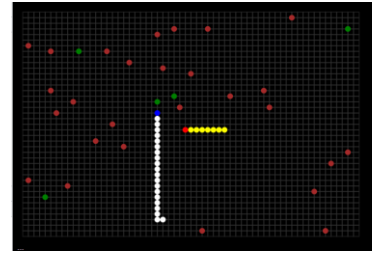
Glizdą poruszamy po planszy za pomocą klawiszy ze strzałkami.

Glizda zjada jabłko: jeżeli zielone, to rośnie jej ogon, jeżeli brązowe to ogon się kurczy.

Glizda może strzelać fragmentem ogona i wtedy strzał zjada wszystko do końca planszy.

Po planszy porusza się również automatyczna glizda, która zjada kolejne jabłko.

Jeżeli jabłka się skończą, losowana jest kolejna partia.



Pamiętaj o tym, by zrzut ekranu DOKUMENTOWAŁ Twoją pracę

Plansza (1)

- W swoim folderze utwórz 2 nowe dokumenty: **js09.html** i **js09.js**
- Otwórz oba dokumenty w notatniku, a dokument HTML w przeglądarce
- Do dokumentu **HTML** wklej tekst z ramki

```
<html>
<head>
  <meta charset=utf8>
  <title> GLIZDA </title>
  <script src=js09.js></script>
</head>
<body style="background-color: black; color: white">
  <canvas width=620 height=420 id=GLIZDA></canvas>

<script>
  var G=GLIZDA.getContext("2d");
  var Gsze=G.canvas.width;
  var Gwys=G.canvas.height;
  G.strokeStyle="white";
  G.strokeRect(0,0,Gsze,Gwys);
  G.fillStyle="white";
  G.fillText("Libront Wacław",0,8);
</script>
</body>
</html>
```

tło strony czarne i kolor znaków biały

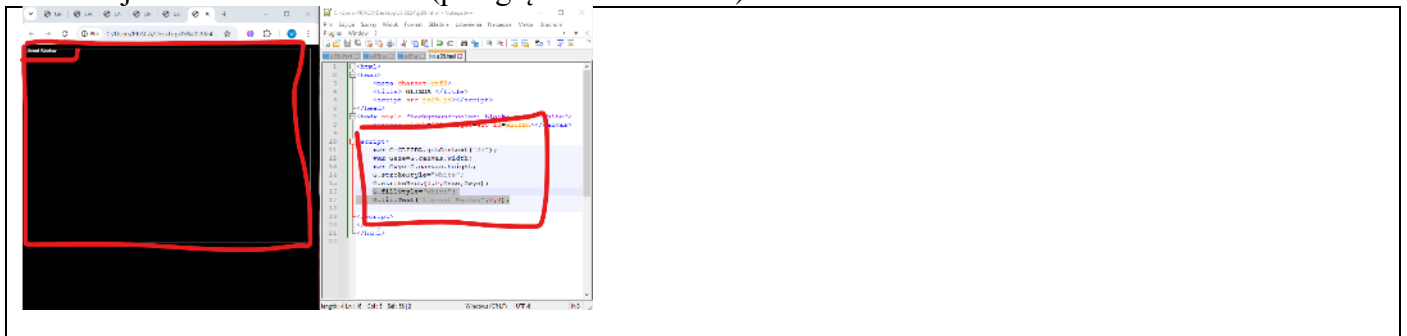
obszar canvas ma rozmiar 620x420 pikseli

dostęp do obszaru za pomocą „uchwyty” o nazwie G

zadeklarowane dwie zmienne Gsze i Gwys, w których zapamiętujemy szerokość i wysokość obszaru canvas

rysujemy białą ramkę na granicach canvas

- Zmień tytuł strony **GLIZDA** na swoje **inicjały**
- Wpisz swoje **nazwisko i imię**
- Zapisz dokumenty i odśwież przeglądarkę
powinna pojawić się ramka o rozmiarze 620x420 pikseli
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Pole gry (1) – szachownica z kwadratów

Plansza, po której porusza się glizda składa się KOLxWIE kwadratów o boku 2*PRO

Ponieważ glizda będzie poruszać się po skrzyżowaniach, więc możliwych pozycji jest KOL+1 (0..60) i WIE+1 (0..40)

Kwadraty rysujemy ciemnoszarym kolorem

- Do dokumentu JS wklej tekst z ramki

```
function FPlansza() {
  var bok=2*PRO;
```

```
G.clearRect(0,0,Gsze,Gwys);
G.strokeStyle="rgb(50,50,50)";
for (var k=0;k<KOL;k++){
    for (var w=0;w<WIE;w++){
        G.strokeRect(bok+k*bok,bok+w*bok,bok,bok);
    }
}
}
```

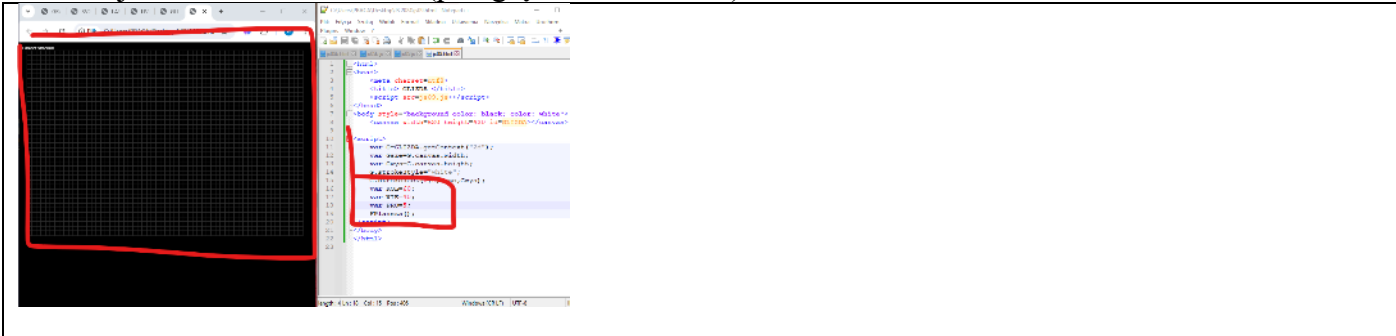
czyścimy obszar canvas
ustalamy ciemnoszary kolor
w pętach FOR rysujemy kwadraty odległe od brzegów o bok, żeby glizda się dobrze rysowała
w lewym górnym rogu wypisujemy nazwisko i imię

- Do dokumentu JS, na koniec funkcji Fplansza } ← przenieś z HTML dwie instrukcje
G.fillStyle="white";
G.fillText("Libront Waclaw",0,8);

- Do dokumentu HTML, przed znacznik </script> ← wpisz instrukcje
var KOL=60;
var WIE=40;
var PRO=5;
Fplansza();

zmienne główne programu: kolumny, wiersze, promień koła (bok=2*promień)

- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Kolorowe koło (jabłko, głowa, ogon) (1)

Kolorowe koła, to główne elementy gry

- Do dokumentu JS wklej tekst z ramki

```
function FKolo(kol,wie,kolor){
    var bok=2*PRO;
    G.beginPath();
    G.fillStyle=kolor;
    G.arc(bok+kol*bok,bok+wie*bok,PRO,0,2*Math.PI);
    G.fill();
}
```

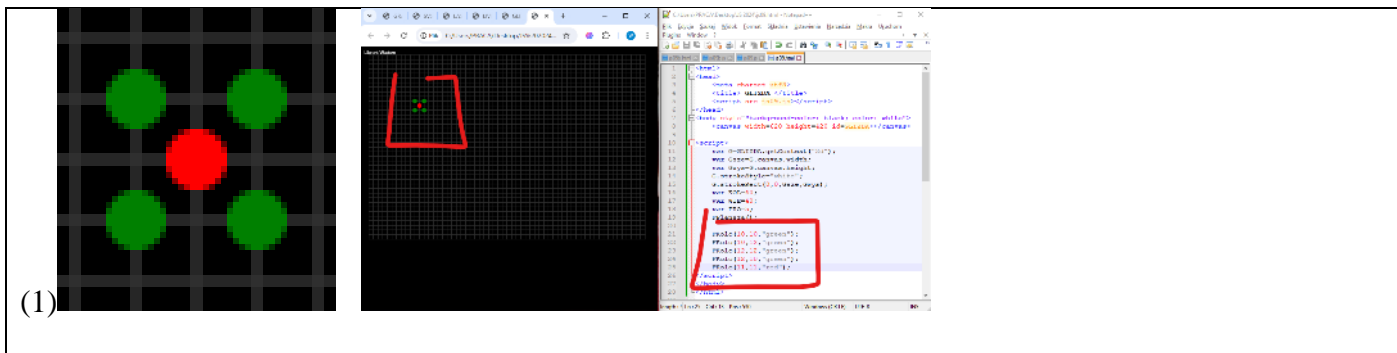
w rogach kwadratów (na skrzyżowaniach) rysowane jest kolorowe koło o promieniu PRO

- Dokument HTML, przed znacznikiem </script> ← wpisz polecenie
FKolo(10,10,"green");

- (1) Dorysuj jeszcze 5 kół, jak na rysunku

```
FKolo(10,10,"green");
FKolo(10,10,"green");
FKolo(10,10,"green");
FKolo(10,10,"red");
```

- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Latająca głowa glizdy (1)

Glizda ma czerwoną głowę, która porusza się w prawo
Gdy glizda osiągnie brzeg planszy, pojawia się po drugiej stronie

- Do dokumentu **HTML**, przed funkcję **Fplansza** `FPlansza();` wpisz zmienne

```
var SKOK=100;
var CZAS;
var Gkol=10;
var Gwie=10;
var GkolV=1;
var GwieV=0;
```

główne zmienne animacji **CZAS** i **SKOK**
współrzędne głowy glizdy (**Gkol**, **Gwie**) i szybkość (**GkolV**, **GwieV**)
glizda porusza się w prawo, bo szybkość kolumnowa=1, a szybkość wierszowa=0

- Do dokumentu **JS** wklej tekst z ramki

```
function FGlizda(){
  FPlansza();
  FKolo(Gkol,Gwie,"red");
  Gkol=Gkol+GkolV;
  Gwie=Gwie+GwieV;
  if (Gkol>KOL) Gkol=0;

  clearTimeout(CZAS);
  setTimeout(FGlizda,SKOK);
}
```

główna pętla animacyjna
rysujemy planszę złożoną z szarych kwadratów
rysujemy czerwoną głowę na skrzyżowaniu (**Gkol**, **Gwie**)
obliczamy nowe współrzędne głowy na podstawie (**GkolV**, **GwieV**)
ponieważ glizda porusza się w prawo (**GkolV**=1), to sprawdzamy przekroczenie prawego brzegu planszy

- Do dokumentu **HTML**, przed znacznik `</script>` wpisz

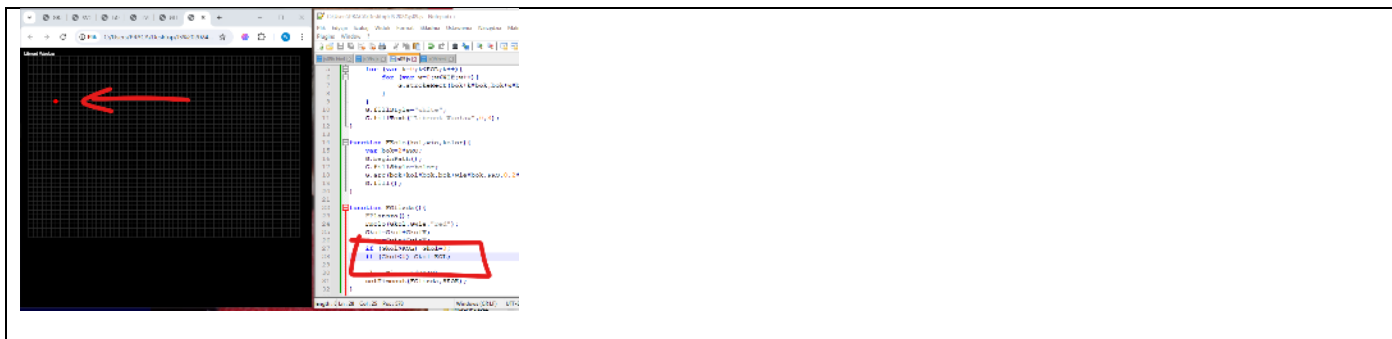
```
FGlizda();
```

uruchomienie pętli animacyjnej

- Glizda porusza się w prawo
- **zmień kierunek** poruszania się w lewo za pomocą zmiennej **GkolV**
glizda porusza się w lewo, ale nie pojawia się na prawym brzegu

```
if (Gkol>KOL) Gkol=0;
clearTimeout(CZAS);
```

- W dokumencie **JS**, wpisz do funkcji **Fglizda()** instrukcję sprawdzającą lewy brzeg
- `if (Gkol<0) Gkol=KOL;`
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Klawiszologia (1)

Do testowania klawiatury i myszki używamy zdarzenia onkeydown.

- Do dokumentu **HTML**, przed znacznikiem `<script>` `<canvas width=` wpisz znaczniki `
` `<label id=idKL></label>` etykieta, w której będzie wypisywany wciśnięty klawisz

- Do dokumentu **HTML**, przed funkcją `FGlizda` `FGlizda()`, wklej tekst z ramki

```
document.addEventListener("keydown", WcisnietyKlawisz, false);
document.addEventListener("keyup", PuszczonyKlawisz, false);
```

obsługa zdarzeń związanych z klawiaturą

w zmiennej `KLA`

jeżeli wciśnięty klawisz, to wywołana zostanie funkcja `WcisnietyKlawisz`

jeżeli puszczono klawisz, to wywołana zostanie funkcja `PuszczonyKlawisz`

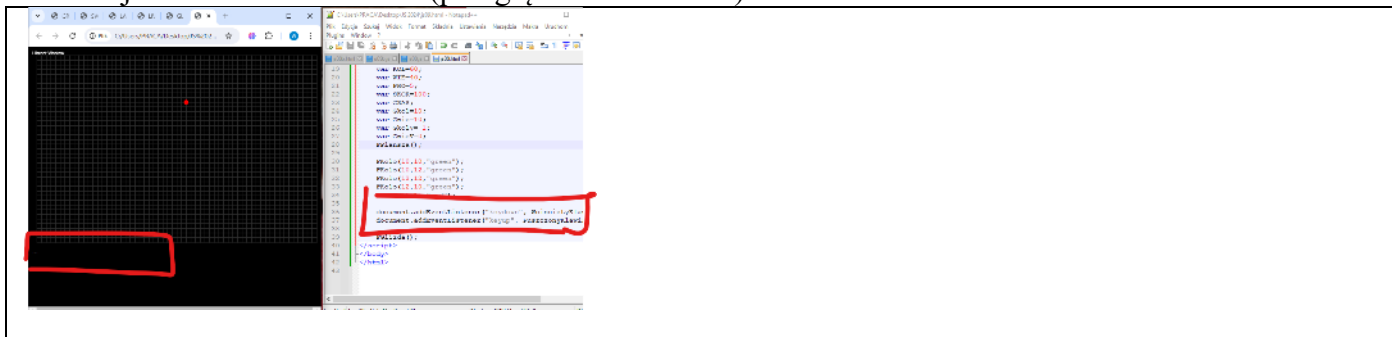
- Do dokumentu **JS** wklej nowe funkcje z ramki

```
function WcisnietyKlawisz(e) {
    var KLA=e.key;
    idKL.innerHTML=KLA;
}
function PuszczonyKlawisz(e) {
    var KLA="...";
    idKL.innerHTML=KLA;
}
```

w zmiennej `KLA` zapamiętujemy opis klawisza

opis klawisza wstawiony to etykiety `idKL`

- „Pobaw się” chwilę klawiaturą i sprawdź opisy klawiszy
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Sterowanie glizdą (1)

Po wciśnięciu klawiszy ze strzałkami, głowa glizdy zaczyna poruszać się w wybranym kierunku, aż do momentu kolejnej zmiany kierunku.

- Do dokumentu **JS**, do funkcji `WcisnietyKlawisz` `idKL.innerHTML=KLA;` wpisz instrukcje


```
if (KLA=="ArrowLeft") {GkolV=-1; GwieV=0;}
if (KLA=="ArrowRight") {GkolV=1; GwieV=0;}
if (KLA=="ArrowUp") {GwieV=-1; GkolV=0;}
if (KLA=="ArrowDown") {GwieV=1; GkolV=0;}
```

jeżeli wciśnięto klawisz ze strzałką, to ustawiamy szybkość tylko w tym kierunku, drugi zerujemy

```
if (Gkol<0) Gkol=KOL;  
clearTimeout(CZAS);
```

- Do dokumentu JS, do funkcji FGlizda

```
if (Gwie>WIE) Gwie=0;  
if (Gwie<0) Gwie=WIE;
```

sprawdzenie prawego i lewego brzegu już powinieneś mieć
dopisz sprawdzanie przejścia przez dolny i górny brzeg
- Ustaw glizdę tak, aby poruszała się wzdłuż dolnej krawędzi planszy
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Losowanie jablek (1)

Poruszająca się glizda będzie zżerać dojrzałe zielone jabłka, co spowoduje przyrost ogona (albo brązowe – zmniejszanie)
Położenie jablek zostanie rozlosowane – za każdym razem 10 zielonych i 10 brązowych
Pozycje jablek powinny znaleźć się w tablicy dwuwymiarowej, która odzwierciedla kratki planszy
W każdej komórce tablicy zapisujemy kolor jabłka; „, ” – bez jabłka

```
document.add
```

- Do dokumentu HTML, przed funkcją FGlizda

```
Fglizda();
```

wpisz instrukcje

```
var JAB=[];  
var IleJab=10;  
FZerujJablka();  
FlosujJablka(IleJab,"green");
```

JAB tablica dwuwymiarowa (po ustawieniu) zawierająca rozlosowane jabłka

IleJab liczba jablek w pojedynczym losowaniu

FZerujJablka funkcja ustawiająca tablicę dwuwymiarową i zerująca wszystkie komórki

FlosujJablka losowanie 10 zielonych jablek

```
Fplansza()
```

- Do dokumentu JS, do funkcji FGlizda

```
FKolo(Gkol,Gwie,"red");
```

wpisz wywołanie funkcji

```
FRysujJablka();
```

FrysujJablka() wrysowanie na planszę jablek, w pętli animacyjnej będzie się automatycznie aktualizować stan jablek

- Do dokumentu JS, wklej tekst z ramki

```
function FZerujJablka() {  
    JAB=[];  
    for (var k=0;k<=KOL;k++){  
        JAB[k]=[];  
        for (var w=0;w<=WIE;w++){  
            JAB[k][w]="";  
        }  
    }  
}  
  
function losowa(p,k) {  
    return Math.floor(Math.random()*(k-p+1)+p);  
}  
  
function FlosujJablka(ile,kolor){  
    var i=ile;  
    do {  
        k=losowa(0,KOL);  
        w=losowa(0,WIE);  
        if (JAB[k][w]=="") {  
            i=i-1;  
        }  
    }  
}
```

```

        JAB[k][w]=kolor;
    }
    } while (i>0)
}

function FRysujJablka() {
    for (var k=0;k<=KOL;k++){
        for (var w=0;w<=WIE;w++){
            var kol=JAB[k][w];
            if (kol!="") FKolo(k,w,kol);
        }
    }
}

```

FZerujJablka

w pierwszej pętli tworzymy kolejne kolumny tablicy JAB – 41 komórek
w drugiej pętli, do w każdej komórce kolumny tworzymy 61 komórek wierszy
do każdej komórki wpisujemy napis pusty „”

FLosowa całkowita liczba losowa z przedziału <p,k>

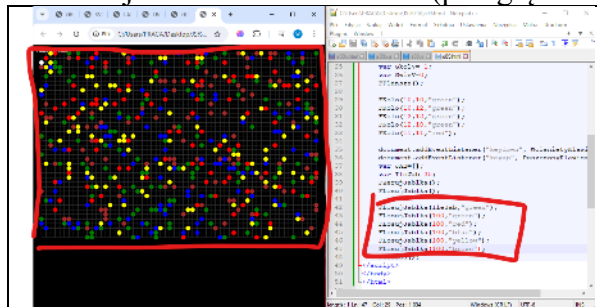
FLosujJablka(ile,kolor)

nie możemy zwykłej pętli for, bo gdy wylosujemy komórkę, w której już jest jabłko? – dlatego pętla logiczna do...while
losujemy dwie współrzędne i sprawdzamy, czy jest pusta
jeżeli tak, to zmniejszamy licznik jabłek do rozlosowania i w komórce wpisujemy kolor
powtarzamy losowanie współrzędnych aż rozlosowane wszystkie jabłka

FRysujJablka

w dwóch pętla FOR przelatujemy przez wszystkie komórki tablicy JAB
jeżeli jest ustawiony kolor (różne od pustego), to rysujemy jabłko w tym kolorze

- Do dokumentu **HTML**, przed funkcję **Fglizda** `Fglizda()`; wpisz polecenie `FLosujJablka(IleJab,"green");`
losuje 20 zielonych jabłek
- Za pomocą tej funkcji rozlosuj jeszcze 6 razy po 100 jabłek w różnych kolorach
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Przycisk losowania (1)

Zautomatyzowane losowanie będzie nam potrzebne w dalszym etapie pracy nad programem

- Do dokumentu JS, wpisz nową funkcję

```

function FUstawJablka() {
    FLosujJablka(IleJab,"green");
    FLosujJablka(IleJab,"brown");
    FRysujJablka()
}

```

- Dokument **HTML**, przed znacznikiem `<script>` `<script>` wpisz

```

<br>
<input type=button value=LOSUJ onClick=FUstawJablka()>

```

przycisk, za pomocą którego rozlosujemy różne kolory jabłek

- W dokumencie HTML

- ustaw zmienną **IleJab** `var IleJab=20;`

- usuń losowanie 6 gatunków jabłek

losowanie zielonych i brązowych jabłek

przerysowanie tablicy z jabłkami

- Kliknij wiele razy przycisk **LOSUJ**, aby rozlosować 300-400 jabłek
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Konsumpcja jabłek (1)

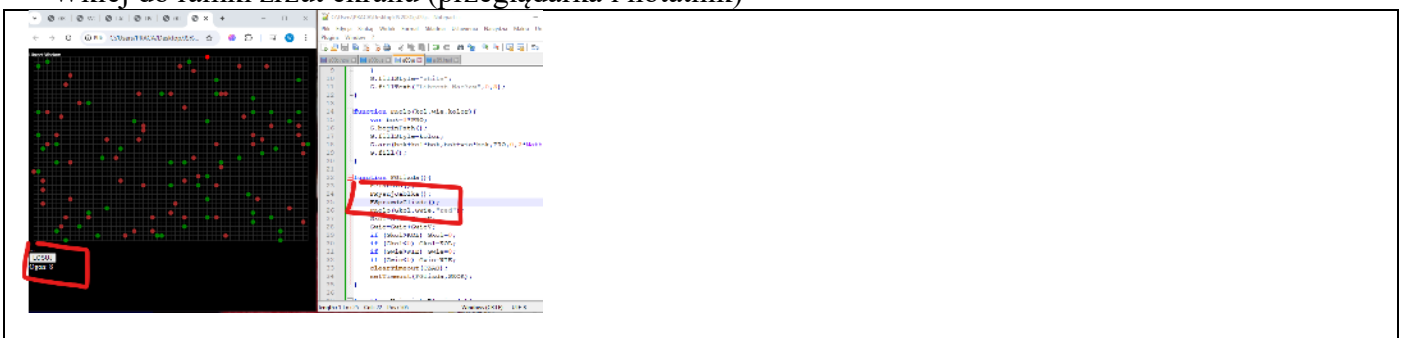
Głowa glizdy wlezie na jablko – zmieniamy długość ogona
 zielone – ogon rośnie, brązowe – ogon maleje
 jeżeli ogon o długości zero, to nie może już zmaleć

- Dokument **HTML**, przed znacznikiem `<script>` `<input type=` `<script>` wpisz `
`
Ogon: `<label id=idOG></label>`
 etykieta, do której wpisujemy aktualną długość ogona
- Do dokumentu **HTML**, przed funkcją **FGlizda**, wpisz definicję zmiennej `var IleOgo=0;`
 zmienna, w której zapisujemy długość ogona, po zjedzeniu jabłka
- Do dokumentu **JS**, wklej tekst z ramki

```
function FSprawdzGlizde() {
    var kol=JAB[Gkol][Gwie];
    switch (kol) {
        case "green":
            JAB[Gkol][Gwie]="";
            IleOgo++;
            break;
        case "brown":
            JAB[Gkol][Gwie]="";
            IleOgo--;
            break;
    }
    idOG.innerHTML=IleOgo;
}
```

pobieramy kolor komórki tablicy, w której znajduje się głowa glizdy
 za pomocą instrukcji warunkowej `switch...case` (rozbudowana wersja `if...else` decydujemy o długości ogona
 zielony kolor: „zerujemy” komórkę tablicy i ogon zwiększa się o jeden
 brązowy kolor: „zerujemy” komórkę tablicy i ogon zmniejsza się o jeden
 na końcu wpisujemy długość ogona do etykiety

- Dokument **JS**, funkcja **FGlizda** `FRysujJablka();` `FKolo(Gkol,Gwie,"red");` wpisz wywołanie funkcji `FSprawdzGlizde();`
 długość ogona sprawdzamy w pętli
- **Spróbuj zjeść 20 jabłek**
 długość ogona 20
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Rysowanie ogona (1)

Ogon, to tablica OGO ze współzrędnymi miejsc, gdzie była głowa.

Po każdym ruchu głowy, pola tablicy OGO przesuwają się o jedno miejsce

```
var IleOgo=0;
FGlizza();
```

- Dokument **HTML**, przed funkcją **FGlizza** wpisz instrukcje

```
var OGO=[];
FUstawJablka();
```

OGO tablica, w której zapisujemy współrzędne członów ogona

FUstawJablka(); po starcie już ustawione jabłka

- Do dokumentu **JS**, wklej tekst z ramki

```
function FUstawOgon(){
    for (var i=IleOgo;i>=0;i--){
        OGO[i-1]=OGO[i-2];
    }
    OGO[0]={k:Gkol,w:Gwie};
}

function FRysujOgon(){
    for (var i=0;i<IleOgo;i++){
        FKolo(OGO[i].k,OGO[i].w,"yellow")
    }
}
```

FUstawOgon za głową podąża ogon, kolejne komórki w OGO muszą zapamiętywać współrzędne głowy

w pętli FOR przesuwamy komórki OGO od tyłu

ostatni komórka ginie

do pierwszej wstawiamy współrzędne głowy

FRysujOgon

w pętli FOR pobieramy współrzędne ogona

i rysujemy kółka w kolorze żółtym

```
FPlansza();
FRysujJablka();
```

- Dokument **JS**, funkcja **FGlizza**, przed funkcją **FRysujJablka** wpisz polecenie

```
FRysujOgon();
```

najpierw rysujemy głowę a potem ogon – w tej kolejności

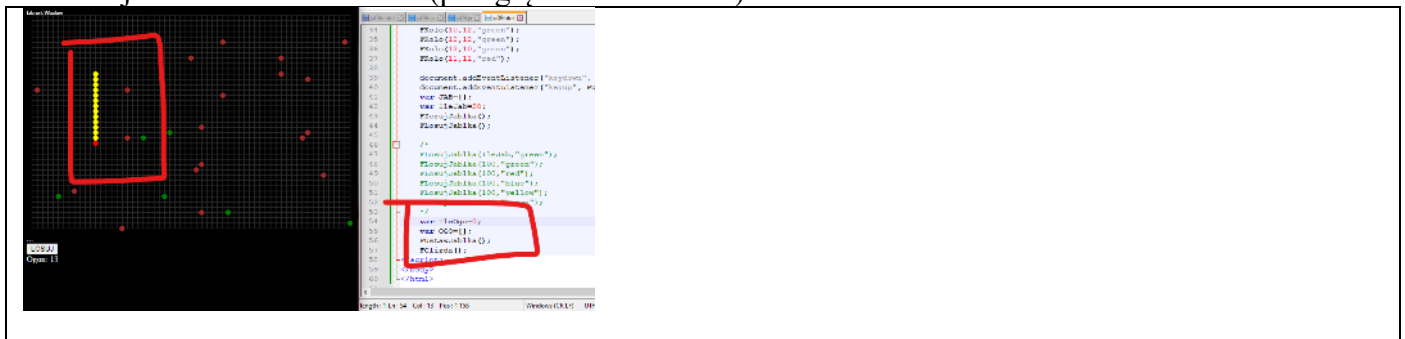
- Dokument **JS**, funkcja **FGlizza**, za funkcją **FSprawdzGlizde** wpisz polecenie

```
FUstawOgon();
```

przestawiamy ogon dopiero po sprawdzeniu trafienia w jabłko – w tej kolejności

- Spróbuj zjeść 20 jabłek – długość ogona 20

- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Glizda strzela ogonem (1)

Wciskamy spację i jeżeli glizda ma ogon, to oddaje strzał.

Fragment ogona leci 3 razy szybciej i kasuje wszystko po drodze.

```
FUstawJablka();
```

- Dokument **HTML**, przed funkcją **FGlizza** wpisz instrukcje


```

var Skol=-2;
var Swie=-2;
var SkolV=0;
var SwieV=0;
var Sszy=3;
var Strz=false;

```

zmiennę, za pomocą których kontrolujemy animację wystrzelonego ogona

(Skol, Swie) pozycja strzału

(SkolV, SwieV) kierunek strzału

Sszy szybkość poruszającego się ogona

Strz=false jeżeli fałsz, to ogon nie leci i można strzelać

jeżeli prawda, to ogon leci i nie można oddać drugiego strzału, albo ogon zerowy i nie ma czym strzelać

```
if (KLA=="ArrowDown")
```

- Dokument JS, funkcja **WcisnietyKlawisz** } **wpisz polecenie**

```
if (KLA==" ") FGlizdaStrzal();
```

jeżeli spacja, to strzelamy ogonem

```
FUstawOgon();
```

- Dokument JS, funkcja **FGlizda**, za funkcją **FUstawOgon** **FKolo**(Gkol, Swie, "red"); **wpisz**

```
FObliczStrzal();
```

sprawdzamy, co zestrzelił pocisk

- Do dokumentu JS, wklej tekst z ramki

```

function FGlizdaStrzal(){
    if (Strz==false){
        if (IleOgo>0) {
            Strz=true;
            IleOgo--;
            Skol=Gkol+GkolV;
            Swie=Gwie+GwieV;
            SkolV=GkolV;
            SwieV=GwieV;
            FKolo(Skol,Swie,"yellow");
        }
    }
}

function FObliczStrzal(){
    for (var i=0;i<Sszy;i++){
        if (Strz==true){
            var kol=JAB[Skol][Swie];
            if (kol!=""){
                switch (kol) {
                    case "green":
                        JAB[Skol][Swie]="";
                        IleOgo++;
                        break;
                    case "brown":
                        JAB[Skol][Swie]="";
                        IleOgo--;
                        break;
                }
                FUstawOgon();
            }
            Skol=Skol+SkolV;
            Swie=Swie+SwieV;
            if (Skol>KOL) Strz=false;
            if (Skol<0) Strz=false;
            if (Swie>WIE) Strz=false;
            if (Swie<0) Strz=false;
            if (Skol>KOL || Skol<0 || Swie>WIE || Swie<0) {
                Strz=false;
                Skol=-2;
                Swie=-2;
            }
        }
    }
    FKolo(Skol,Swie,"yellow");
}

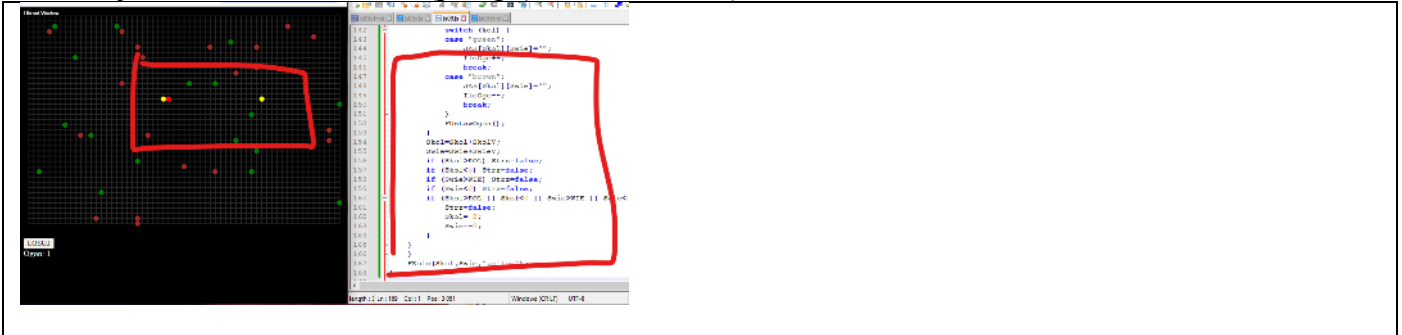
```

FGlizdaStrzal *jeżeli można strzelać (jest ogon i nie wystrzelono wcześniej) to*
Strz=true *od tego momentu nie moa strzelać, aż pocisk wyleci za planszę*

IleOgo-- zmniejszamy ogon o jeden
ustalamy współrzędne i szybkość pocisku – takie jak głowa glizdy
rysujemy żółte koło pocisku

FObliczStrzal sprawdzamy, co po drodze zestrzelił pocisk
pocisk leci trzy razy szybciej niż automat glizdy, dlatego trzeba w pętli analizować trzy ruchu pocisku do przodu
kol != "" jeżeli pocisk trafił na jabłko
switch sprawdzamy co to za jabłko i dodajemy lub ujmujemy z ogona
i wywołujemy funkcję FUstawOgon, która przestawia współrzędne ogona
zmieniamy współrzędne pocisku o jeden i sprawdzamy czy pocisk za ekranem
a jeżeli poza ekranem, to nie można strzelać i żeby go nie było widać na ekranie, to współrzędne (-2,-2)

- Strzel pociskiem i zjedz jabłko
- Wklej do ramki rzut ekranu (przeglądarka i notatnik)



Automat – ruch (1)

Czy można grać z komputerowym „myślącym” przeciwnikiem?
W pierwszym etapie uruchomimy drugą automatyczną glizdę. W kolejnym nauczymy ją myśleć.

- Dokument **HTML**, przed funkcją **FGlizda** `FGlizda();` wpisz deklaracje zmiennych

```
var Akol=50;
var Awie=30;
var AkolV=-1;
var AwieV=0;
var IleAut=0;
var AUT=[];
```

zmiennie odpowiedzialne za ruch automatycznej glizdy
nasza automatyczna glizda porusza się na razie cały czas w lewo

- Dokument **HTML**, za etykietą **idOG** `<script>` wpisz znaczniki

```
Auto: <label id=idAU></label>
```

tutaj będzie widać długość ogona automatycznej glizdy

- Dokument **JS**, funkcja **FGlizda**, przed funkcją **clearTimeout** `clearTimeout(CZAS);` wpisz `Automat();`

wszystkie obliczenia związane z automatyczną glizdą w pętli głównej programy

- Do dokumentu JS, wklej tekst z ramki

```
function Automat(){
    FKolo(Akol,Awie,"blue");
    //Rysuj Ogon
    for (var i=0;i<IleAut;i++){
        FKolo(AUT[i].k,AUT[i].w,"white")
    }
    //Sprawdz w co trafiła głowa
    if (JAB[Akol][Awie]=="green"){
        JAB[Akol][Awie]="";
        IleAut++;
        AUT.push({k:0,w:0});
    }
    if (JAB[Akol][Awie]=="brown"){
        JAB[Akol][Awie]="";
        if (IleAut>0) {
            IleAut--;
            AUT.slice(AUT.length-1);
        }
    }
}
```

```

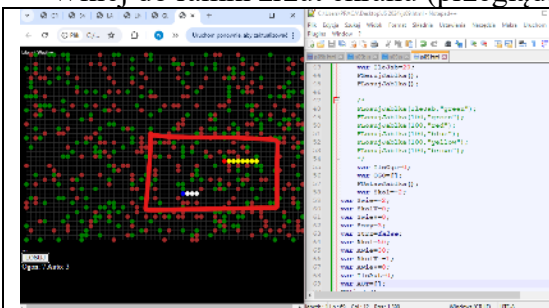
    }
  }
  idAU.innerHTML=IleAut;
  //Przestaw Ogon
  for (var i=IleAut;i>=0;i--){
    AUT[i-1]=AUT[i-2];
  }
  AUT[0]={k:Akol,w:Awie};

  //ruch głowy
  Akol=Akol+AkolV;
  Awie=Awie+AwieV;
  //brzegi
  if (Akol>KOL) Akol=0;
  if (Akol<0) Akol=KOL;
  if (Awie>WIE) Awie=0;
  if (Awie<0) Awie=WIE;
}

```

Wszystkie niezbędne operacje w jednej funkcji
 rysowana jest niebieska głowa automatu
 rysujemy biały ogon
 sprawdzamy, czy głowa nie zżarła jabłka
 wpisujemy długość ogona do etykiety
 przestawiamy tablicę ogona
 obliczamy nowe współrzędne głowy
 sprawdzamy przejścia przez brzeg planszy

- Umieść na planszy mnóstwo jabłek, aby automatyczna glizda miała ogon
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Glizda myśli (1)

Glizda szuka najbliższego jabłka - odległość do niego wylicza z tw. Pitagorasa
 W dalszej kolejności sprawdza, który kierunek jest krótszy i ten wybiera, chyba, że już jest na tym samym pionie lub poziomie
 Jeżeli zero jabłek na tablicy, to automat losuje kolejne

//ruch głowy

- Dokument JS, funkcja **Automat**, przed instrukcjami `Akol=Akol+AkolV;` wpisz polecenie

`AUustalKierunek();`

instrukcje wyliczają pozycję głowy

wszystkie obliczenia związane z automatyczną glizdą w pętli głównej programy

- Dokument JS, wklej tekst z ramki

```

function ASzukajJablka(Ak, Aw) {
  var min={k:0,w:0,odl:999}
  for (var k=0;k<KOL;k++){
    for (var w=0;w<WIE;w++){
      if (JAB[k][w]=="green"){
        var dk=Ak-k;
        var dw=Aw-w;
        var odl=Math.sqrt(dk*dk+dw*dw);
        if (odl<min.odl) {
          min.k=k;
          min.w=w;
          min.odl=odl;
        }
      }
    }
  }
  return min;
}

```

```

}

function AUstalKierunek(){
  var min=ASzukajJablka(Akol,Awie);
  if (min.odl<999){
    var dk=Akol-min.k;
    var dw=Awie-min.w;
    if (Math.abs(dk) < Math.abs(dw)) {
      if (Math.abs(dk)>0 ){
        AkolV=Math.sign(-dk);AwieV=0;
      } else {
        AwieV=Math.sign(-dw);AkolV=0;
      }
    }
  } else {
    if (Math.abs(dw)>0 ) {
      AwieV=Math.sign(-dw);AkolV=0;
    } else {
      AkolV=Math.sign(-dk);AwieV=0;
    }
  }
  } else { //gdym jest co szukac
    FUstawJablka();
  }
}
}

```

ASzukajJablka szuka najbliższego jabłka

min w zmiennej zapisana pozycja i odległość do najbliższego jabłka od głowy glizdy
 w pętach FOR szukamy jabłek i wyliczamy odległość do jabłka
 jeżeli wyliczona odległość jest mniejsza niż zapisana w min, to zamieniamy
 w efekcie po zakończeniu sprawdzania, w min będzie najbliższe jabłko
 funkcja zwraca tablicę min

AUstalKierunek w jakim kierunku ma iść glizda, żeby dojść do jabłka
 najpierw szukamy najbliższego jabłka za pomocą ASzukajJablko
 wyliczamy odległości w pionie i poziomie do jabłka
 wybieramy kierunek ten z mniejszą odległością
 ale nie może być na tym samym poziomie (pionie)
 a gdy nie ma czego szukać, bo brak jabłek, to losowanie nowych

- Spróbuj powalczyć z automatyczną glizdą i zbierz 20 jabłek w ognie
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

