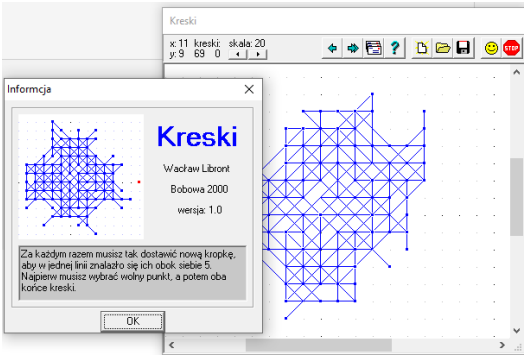
JS 10 - Kreski (13)



KRECHY – gra dla znudzonych uczniów z czasów, gdy nie było komputerów i trzeba było czymś zająć czas na lekcji. Wystarczy pokratkowana kartka papieru i długopis.
Początkowy układ kropek tworzy „krzyż". Za każdym razem należy tak dostawić nową kropkę, aby w jednej linii znalazło się ich obok siebie 5. W grze najpierw musisz wybrać wolny punkt, a potem wskazać oba końce kreski. Gra kończy się, gdy nie masz możliwości postawienia nowej kreski. Komu uda pobić się szkolny rekord i narysować jak najwięcej kresek?

Pamiętaj o tym, by zrzut ekranu DOKUMENTOWAŁ Twoją pracę

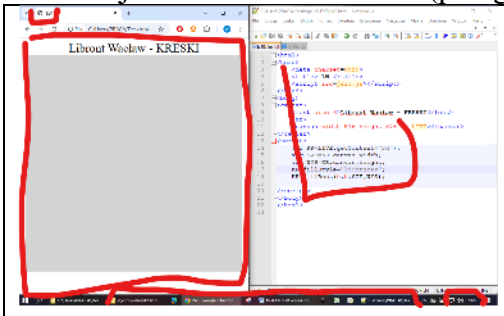
1) Pliki

- W swoim folderze utwórz 2 nowe dokumenty: **js13.html** i **js13.js**
- Otwórz oba dokumenty w notatniku, a dokument HTML w przeglądarce
- Ustaw oba okna na dwóch połowach monitora
- Dokument **HTML** - wklej tekst z ramki

```
<html>
<head>
	<meta charset=utf8>
	<title> KRESKI </title>
	<script src=js10.js></script>
</head>
<body>
<center>
	<font size=6>Libront Wacław - KRESKI</font>
	<br>
	<canvas width=576 height=576 id=LIVE></canvas>
</center>
<script>
	var KR=LIVE.getContext("2d");
	var SZE=KR.canvas.width;
	var WYS=KR.canvas.height;
	KR.fillStyle="black";
	KR.fillRect(0,0,SZE,WYS);

</script>
</body>
</html>
```

obszar canvas ma rozmiar 576x576 pikseli
dostęp do canvas za pomocą „uchwytu" o nazwie KR
zadeklarowane dwie zmienne SZE i WYS, w których zapamiętujemy szerokość i wysokość obszaru canvas
wypełniamy canvas czarny kolorem

- Wpisz swoje **nazwisko i imię**
- Zmień kolor canvas na **lightgrey**
- Zapisz dokumenty i odśwież przeglądarkę
  Szary kwadrat
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



2) Kratki

- Dokument **JS** - wklej nowe funkcje z ramki

```
function Kratka(x,y){
	KR.strokeStyle=RAM;
	KR.lineWidth=LIN;
	KR.strokeRect(x,y,BOK,BOK);
}
function RysujZeszyt(){
	for (var w=0;w<ILE;w++)
	for (var k=0;k<ILE;k++){
		var x=k*BOK;
		var y=w*BOK;
		Kratka(x,y);
	}
}
```

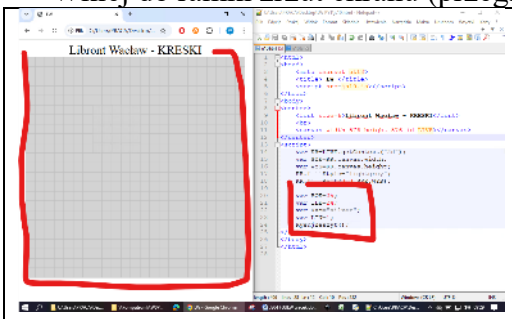
Funkcja Ramka() = rysowanie kwadratowej ramk1 - lewy górny róg (x,y)
Funkcja RysujZeszyt() - rysowanie ILE ramek (w pionie i poziomie) o wysokości i szerokości BOK

```
	KR.fillRect(0,0,SZE,WYS);
</script>
```

- Dokument **HTML** wpisz polecenia:

```
var BOK=25;
var ILE=24;
var RAM="silver";
var LIN=1;
RysujZeszyt();
```

- Zapisz dokumenty i odśwież przeglądarkę
  Szary kwadrat z kratkami
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



3) Kropki

Kropki stawiamy na przecięciu pionowych i poziomych linii.
Jest ich 24 i ponumerowane są od 0 do 23
Położenie kropek pamiętane będzie w tablicy KROPKI

- Dokument **JS** – wklej nowe funkcje z ramki

```
function Kropka(k,w,pro,kol){
	KR.beginPath();
	var x=k*BOK;
	var y=w*BOK;
	KR.arc(x,y,pro,0,2 * Math.PI);
	KR.fillStyle=kol;
	KR.fill();
	KR.closePath();
}

function ZerujKropki(){
	for (var k=0;k<ILE;k++){
		KROPKI[k]=[];
		for (var w=0;w<ILE;w++){
			KROPKI[k][w]=0;
		}
	}
}

function RysujKropki(){
	var pro=6;
	for (var k=0;k<ILE;k++)
	for (var w=0;w<ILE;w++){
```

```
            if (KROPKI[k][w]==1)
                  Kropka(k,w,pro,"blue");
            if (KROPKI[k][w]==0)
                  Kropka(k,w,1,"grey");
      }
}
```

Funkcja KROPKA - rysowanie kropek o promieniu PRO i kolorze KOL
K i W to numer kolumny i wiersza, na przecięciu których narysuje się kropka
Funkcja ZERUJKROPKI ustawia wszystkie komórki tablicy KROPKI na zero
Funkcja RYSUJKROPKI sprawdza tablicę KROPKI
jeżeli komórka jest równa zero, to rysowana jest mała szara kropka
w przeciwnym razie rysowana będzie niebieska duża kropka

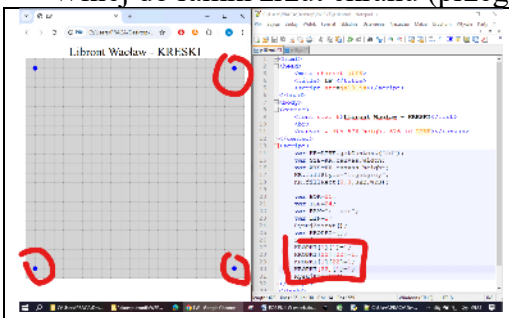
```
      RysujZeszyt();
      [redacted]
</script>
```

- Dokument **HTML** wpisz polecenia:

```
var KROPKI=[];
ZerujKropki();
KROPKI[1][1]=1;
KROPKI[redacted]=1;
KROPKI[redacted]=1;
KROPKI[redacted]=1;
RysujKropki();
```

W tablicy KROPKI wstaw liczbę 1 (jeden), a na planszy pojawi się duża, niebieska kropka

- Ustaw jeszcze **trzy niebieskie kropki** w rogach planszy
- Zapisz dokumenty i odśwież przeglądarkę
  W rogach planszy, na przecięciu linii, niebieskie kropki
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



4) Plansza

Początkowy układ kropek na planszy, to „krzyż”

- Dokument **JS** - wklej nowe funkcje z ramki

```
function UstawKrzyz(){
      KROPKI[10][14]=1; KROPKI[10][15]=1;     KROPKI[10][16]=1;
      KROPKI[11][16]=1; KROPKI[12][16]=1;     KROPKI[13][16]=1;
      KROPKI[13][15]=1; KROPKI[13][14]=1;     KROPKI[1][ 1]=1;
      KROPKI[14][13]=1; KROPKI[15][13]=1;     KROPKI[16][13]=1;
      KROPKI[16][12]=1; KROPKI[16][11]=1;     KROPKI[16][10]=1;
      KROPKI[15][10]=1; KROPKI[14][10]=1;     KROPKI[22][ 1]=1;
      KROPKI[13][ 9]=1; KROPKI[13][ 8]=1;     KROPKI[13][ 7]=1;
      KROPKI[12][ 7]=1; KROPKI[11][ 7]=1;     KROPKI[10][ 7]=1;
      KROPKI[10][ 8]=1; KROPKI[10][ 9]=1;     KROPKI[ 1][22]=1;
      KROPKI[ 9][10]=1; KROPKI[ 8][10]=1;     KROPKI[ 7][10]=1;
      KROPKI[ 7][11]=1; KROPKI[ 7][12]=1;     KROPKI[ 7][13]=1;
      KROPKI[ 8][13]=1; KROPKI[ 9][13]=1;     KROPKI[22][22]=1;
}

function RysujPlansze(){
      KR.fillStyle="lightgrey";
      KR.fillRect(0,0,SZE,WYS);
      RysujZeszyt();
      RysujKropki();
}
```

Funkcja UstawKrzyz ustawia komórki tablicy odpowiedzialne za podstawowy „krzyż”
Funkcja RysujPlansza przerysowuje planszę od początku

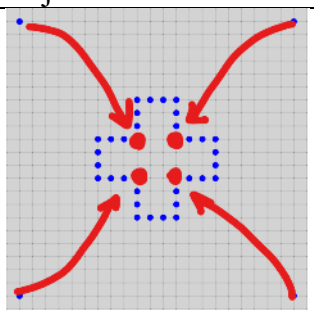
```
RysujKropki();
```
</script>

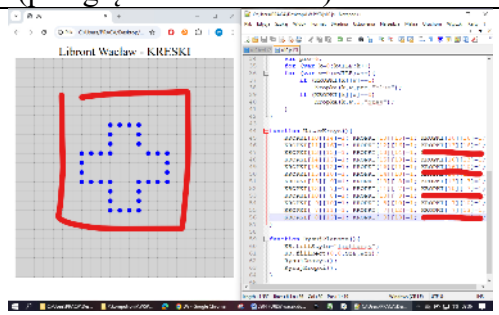
- Dokument **HTML** wpisz polecenia

```
ZerujKropki();
UstawKrzyz();
RysujPlansze();
```

- (1) Cztery kropki „uciekły” ze swoich właściwych położeń
- Dokument **JS**, funkcja **UstawKrzyz** - popraw współrzędne tablicy **KROPKI** aby kropki znalazły się w środku
- Zapisz dokumenty i odśwież przeglądarkę
  „Krzyż” z niebieskich kropek - początkowe ustawienie
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

(1)





5) Myszka

Grę obsługujemy za pomocą myszki klikając w pola planszy
Kropki wstawiamy na przecięciach kolumn i wierszy (jasne linie – szare kropki)
Musimy „nauczyć” program przeliczać współrzędne myszki na kratki i na odwrót

```
<canvas width=576 h
```

- Dokument **HTML** - wpisz polecenia

```
<br><br>
<label id=idMYSZ> </label>
```

ramka, w której pojawią się współrzędne wskaźnika myszki

```
RysujPlansze();
```
</script>

- Dokument **HTML** - wpisz polecenia

```
var MYSZKA={x:0,y:0};
MyszkaNapis();
Zdarzenia();
```

zmienna MYSZKA zawiera da pola {x,y}, w których zapamiętujemy współrzędne wskaźnika myszki w obszarze canvas
funkcja MyszkaNapis wstawia do pola idMYSZ współrzędne
funkcja Zdarzenia uruchamia obsługę myszki

- Dokument **JS** wklej nowe funkcje z ramki

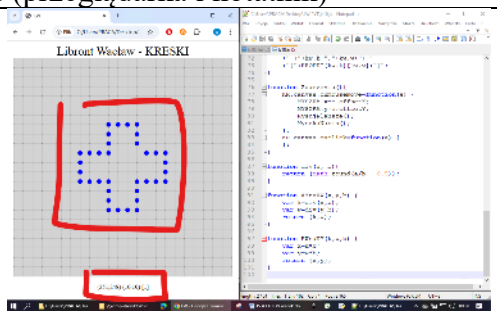
```
function MyszkaNapis(){
	var x=MYSZKA.x;
	var y=MYSZKA.y
	var kw=XYnaKW(x+BOK/2,y+BOK/2,BOK)
	idMYSZ.innerHTML=
	"("+x+","+y+")"
	+" ("+kw.k+","+kw.w+") "
	+"["+KROPKI[kw.k][kw.w]+"]";
}

function Zdarzenia(){
	KR.canvas.onmousemove=function(e) {
		MYSZKA.x=e.offsetX;
		MYSZKA.y=e.offsetY;
		RysujPlansze();
		MyszkaNapis();
	};
	KR.canvas.onclick=function(e) {
	};
}
```

```
function div(a, b){
	return (Math.round(a/b - 0.5));
}

function XYnaKW(x,y,b) {
	var k=div(x,b);
	var w=div(y,b);
	return {k,w};
}

function KWnaXY(k,w,b) {
	var x=k*b;
	var y=w*b;
	return {x,y};
}
```

funkcja MyszkaNapis wstawia do etykiety idMYSZ napis złożony ze współrzędnych myszki
funkcja Zdarzenia obsługuje dwa zdarzenia związane z myszką: przesuwanie i klikanie
polecenie e.offset sprawdza położenie myszki w obszarze canvas i przypisuje do zmiennej MYSZKA
funkcje XYnaKW i KWnaXY przeliczają współrzędne - wykorzystują dzielenie całkowite div

- Zapisz dokumenty i odśwież przeglądarkę
  Wskaźnik myszki pokazuje współrzędne
- (1) Ustaw wskaźnik myszki w kropce **{10,10}**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

(251,249) (10,10) [1]
(1)



6) Klikanie - wstawianie kropek

Aby ustawić kropkę, wystarczy w tablicy KROPKI wstawić do odpowiedniej komórki jedynkę i przerysować planszę
Podczas przesuwania myszki po planszy możemy ustawiać w odpowiedni sposób wskaźnik, który będzie pokazywał wybraną kropkę w odpowiednim kolorze

- Dokument **HTML** wpisz znaczniki

```
myszka <label id=idMYSZ> </label>
</center>
<br>
kropki <label id=idRUCH> </label>
```

- Dokument **JS** - wklej nowe funkcje z ramki

```
function UstawKropke(){
	var x=MYSZKA.x;
	var y=MYSZKA.y
	var kw=XYnaKW(x+BOK/2,y+BOK/2,BOK)
	var k=kw.k;
	var w=kw.w;
	KROPKI[k][w]=1;
}

function IleKropek(){
	var ile=0;
	for (var k=0;k<ILE;k++)
	for (var w=0;w<ILE;w++)
		if (KROPKI[k][w]==1) ile++;
	return ile;
}

function RysujWskaznik(){
	var x=MYSZKA.x;
	var y=MYSZKA.y
	var kw=XYnaKW(x+BOK/2,y+BOK/2,BOK);
	var k=kw.k;
```

```
        var w=kw.w;
        var pro=10;
        if (KROPKI[k][w]==1)
              Kropka(k,w,pro,"red");
        if (KROPKI[k][w]==0)
              Kropka(k,w,pro,"aqua");
}
```

Funkcja USTAWKROPKE przeliczamy współrzędne myszki na kolumny i wiersze i ustawiam tablicę w komórce (k,w)
Funkcja ILEKROPEK zlicza jedynki w tablicy KROPKI
Funkcja RYSUJWSKAZNIK - wskaźnik gracza jest kropką, która w zależności od zawartości tablicy KROPKI przyjmuje różne kolory

- Dokument **JS**, funkcja **MyszkaNapis**, `idRUCH.innerHTML=IleKropek()` - wklej polecenia

```
+"["+KROPKI[kw.k][kw.w]+"]";
}
```

- Dokument **JS**, funkcja **Zdarzenia**, zdarzenie **onmousemove** `RysujWskaznik();` - wpisz

```
RysujPlansze();
MyszkaNapis();
```

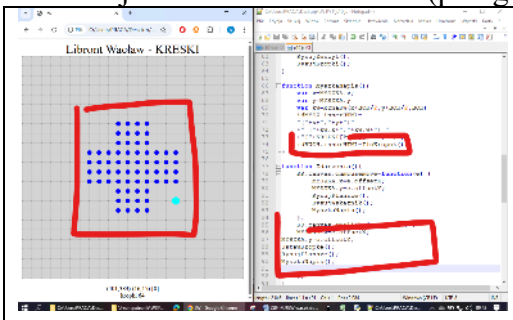
- Dokument **JS**, funkcja **Zdarzenia**, zdarzenie **onclick** - wklej polecenia

```
KR.canvas.onclick=function(e) {
};
```

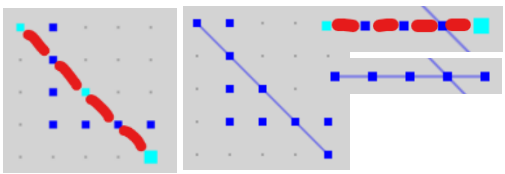
```
MYSZKA.x=e.offsetX;
MYSZKA.y=e.offsetY;
UstawKropke();
RysujPlansze();
MyszkaNapis();
```

kliknięto w obszar canvas - przeliczamy współrzędne myszki na kolumny i wiersze
ustawiamy tablicę w komórce (k,w) i przerysowujemy planszę

- Zapisz dokumenty i odśwież przeglądarkę
  Za pomocą klikania można wstawiać niebieskie kropki
- Klikaj w „puste” kratki i wypełnij „krzyż” kropkami
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



7) Ruch



Aby ustawić kreskę należy:
1) wstawić kropkę na puste pole
2) zaznaczyć początek kreski na kropce (może być przed chwilą wstawiona)
3) zaznaczyć koniec kreski na kropce (może być przed chwilą wstawiona)
Poszczególne etapy ruchu zapamiętamy w tablicy RUCH
[0] – etap stawiania kropek
*[1,2] – kolumna i wiersz pierwszej kropki (na **pustym** polu)*
*[3,4] – kolumna i wiersz na **początku** kreski (na ustawionej już kropce)*
*[5,6] – kolumna i wiersz na **końcu** kreski (na ustawionej już kropce)*

Komórki tablicy KROPKI mogą teraz przyjmować wartości:
0 – nie ma kropki
1 – kropka ustawiona
10 – pierwsza kropka
11,21 – druga lub trzecia kropka, postawiona na kropce
20 – druga lub trzecia kropka, postawiona na pierwszej kropce

- Dokument **HTML** - wpisz polecenia

```
var MYSZKA={x:0,y:0};
MyszkaNapis();
```

```
var RUCH=[7];
for (var i=0;i<7;i++) RUCH[i]=0;
```

tablica RUCH, w której zapamiętamy poszczególne etapy stawiania kropek (kreski)

- Dokument **JS,** funkcja **MyszkaNapis()** - wklej tekst z ramki

```
idRUCH.innerHTML=IleKropek()
}
```

```
+" "+RUCH[0]
+"("+RUCH[1]+","+RUCH[2]+")"
+"("+RUCH[3]+","+RUCH[4]+")"
+"("+RUCH[5]+","+RUCH[6]+")";
```

Ustawiamy kropki – poszczególne etapy i współrzędne kropek pokazują się w polach

- Dokument **JS,** funkcja **Zdarzenia**, zdarzenie **onclick**

  - **usuń** polecenie `UstawKropke();`

  - **wpisz** polecenie `Ruch();`

po każdym kliknięciu analizujemy etapy stawiania kropek

- Dokument **JS,** funkcja **RysujWskaznik()** - wklej tekst

```
Kropka(k,w,bok,"aqua");
}
```

```
if (RUCH[0]>0){
	if (KROPKI[k][w]>0)
		Kropka(k,w,pro,"aqua");
	if (KROPKI[k][w]==0)
		Kropka(k,w,pro,"red");
}
```

gdy ustawiamy kropki kresek, to kolory wskaźnika odwrotnie – końce kresek już na ustawionych kropkach

- Dokument **JS,** funkcja **RysujKropki()** - wpisz tekst z ramki

```
Kropka(k,w,1,"grey");
	}
}
```

```
if (KROPKI[k][w]>1)
    Kropka(k,w,pro,"aqua");
```

Gdy w tablicy KROPKI coś więcej niż jeden, to rysuj na aqua – pierwsza kropka nowej kreski
Gdy kreska ustawiona poprawnie, to ta kropka zamieni się na prawdziwą z wartością 1

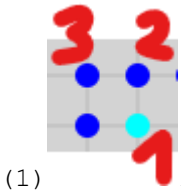
- Dokument **JS** - wklej nowe funkcje z ramki

```
function Ruch(){
	var x=MYSZKA.x;
	var y=MYSZKA.y
	var kw=XYnaKW(x+BOK/2,y+BOK/2,BOK);
	var k=kw.k;
	var w=kw.w;
	if (RUCH[0]==0 && KROPKI[k][w]==0){
		RUCH[0]=1;
		KROPKI[k][w]=KROPKI[k][w]+10;
		RUCH[1]=k;
		RUCH[2]=w;
	}
	else
	if (RUCH[0]==1 && KROPKI[k][w]>0){
		RUCH[0]=2;
		KROPKI[k][w]=KROPKI[k][w]+10;
		RUCH[3]=k;
		RUCH[4]=w;
	}
	else
	if (RUCH[0]==2 && KROPKI[k][w]>0){
		RUCH[0]=3;
		KROPKI[k][w]=KROPKI[k][w]+10;
		RUCH[5]=k;
		RUCH[6]=w;
	}
	if (RUCH[0]==3){
		KROPKI[RUCH[1]][RUCH[2]]=1;
		ZerujRuch();
	}
}
```

```
function ZerujRuch(){
	if (RUCH[0]>0){
		var k0=RUCH[1];
		var w0=RUCH[2];
		var k1=RUCH[3];
		var w1=RUCH[4];
		var k2=RUCH[5];
		var w2=RUCH[6];
		KROPKI[k0][w0]=KROPKI[k0][w0] % 10;
		KROPKI[k1][w1]=KROPKI[k1][w1] % 10;
		KROPKI[k2][w2]=KROPKI[k2][w2] % 10;
	}
	RUCH[0]=0;
}
```

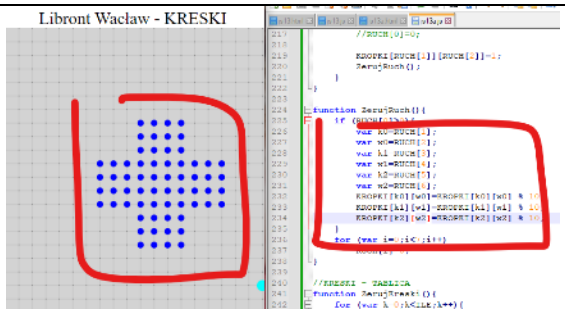
Klikamy w planszę
*Jeżeli to **pierwsza** kropka, to musi być na pustym polu*
Komórka w KROPKI przyjmuje wartość 10
RUCH[0] przyjmuje wartość 1
Zapamiętujemy współrzędne w RUCH[1] i RUCH[2]
*W przeciwnym razie, gdy **druga** kropka i pole już zajęte (może być nawet zajęte pierwszą kropką)*
Komórka w KROPKI jest powiększana o 10
(może być 11 lub 21, gdy była tam kropka lub 20, gdy ustawiliśmy tam pierwszą kropkę)
Wykonujemy podobne czynności jak przy pierwszej kropce
*W przeciwnym razie, gdy **trzecia** kropka i pole już zajęte (może być nawet zajęte pierwszą kropką)*
Komórka w KROPKI jest powiększana o 10
(może być 11, 21 lub 31, gdy była tam kropka lub 30, gdy ustawiliśmy tam pierwszą kropkę)
Wykonujemy podobne czynności jak przy pierwszej kropce
Jeżeli ustawiono już trzy kropki, to
Zerujemy ruch, aby można było ustawiać kolejną kreskę

- Zapisz dokumenty i odśwież przeglądarkę
  Klikanie powoduje wstawianie niebieskich kropek
- (1) Wypełnij „krzyż” **niebieskimi** kropkami
  - wybierz w puste pole
  - wybierz dwa pola zajęte
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



(1) lub


lub


itp.


8) Poprawny ruch

Aby poprawnie wstawić kreskę należy:
1) wstawić kropkę na puste pole
2) zaznaczyć początek kreski na ustawionej kropce (może być przed chwilą wstawiona)
3) zaznaczyć koniec kreski na ustawionej kropce (może być przed chwilą wstawiona)
UWAGI
- początek i koniec kreski muszą leżeć na tej samej prostej
- odległość między początkiem i końcem kreski musi wynosić dokładnie 5 kropek
- nowa kropka może być ustawiona w dowolnym punkcie kreski
- kreska nie może zawierać kratek bez kropek
- kreski mogą się przecinać, ale nie mogą się nakładać
- kreski mogą się łączyć końcami
Te wszystkie „uwagi” należy zamienić na język algorytmiki

- Dokument **JS** - wklej nową funkcję z ramki

```
function CzyPoprawnyRuch(){
	var k0=RUCH[1];
	var w0=RUCH[2];
	var k1=RUCH[3];
```

```
        var w1=RUCH[4];
        var k2=RUCH[5];
        var w2=RUCH[6];
        var dk=k2-k1;
        var dw=w2-w1;
        var OK1=true;
        if (dk==0 && dw==0) OK1=false;
        var OK4=false;
        var dk4=Math.abs(dk);
        var dw4=Math.abs(dw);
        if ((dk4==4 || dk4==0) && (dw4==4 || dw4==0)) OK4=true;
        if (dk!=0) dk=div(dk,Math.abs(dk));
        if (dw!=0) dw=div(dw,Math.abs(dw));
        var OK0=false
        var OK2=false;
        var OK3=true
        for (var i=0;i<5;i++){
               var ki=k1+dk*i;
               var wi=w1+dw*i;
               if (ki==k0 && wi==w0) OK0=true;
               if (ki==k2 && wi==w2) OK2=true;
               if (KROPKI[ki][wi]==0) OK3=false;
        }
        var OK=OK0 & OK1 & OK2 & OK3 & OK4;
        return OK;
}
```

Funkcja sprawdza czy trzy ustawione kropki spełniają kryteria poprawności
- *czy nie kliknięto w tym samym miejscu – OK1*
- *odległość pomiędzy nimi wynosi dokładnie 4 - OK4*
- *czy nie ma pomiędzy nimi pustych pól - OK3*
- *czy początkowe pole jest w linii – OK0*
- *czy końcowa kropka jest w linii – OK2*

Gdy iloczyn logiczny OK0, OK1, OK2, OK3, OK4 jest prawdziwy, to można wstawiać kreskę

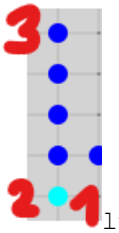
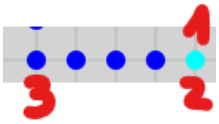
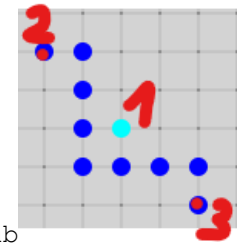
```
if (RUCH[0]==3){
    KROPKI[RUCH[1]][RUCH[2]]=1;
```

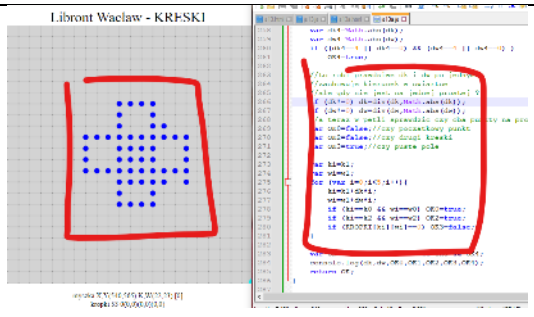
- Dokument **JS,** funkcja **Ruch()**
  - zastąp zaznaczony wiersz nową instrukcją warunkową

```
if (CzyPoprawnyRuch()==true){
     KROPKI[RUCH[1]][RUCH[2]]=1;
}
```

Gdy poprawny ruch, to wstaw do tablicy KROPKI 1 w początkowym punkcie
Po przerysowaniu tablicy, nowa kropka pojawi się na planszy
Pozostałe kropki zostaną wyzerowane
Gdy wykonamy ruch zły to wszystkie ustawione kropki będą wyzerowane

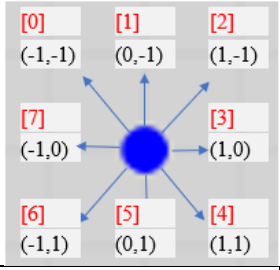
- Zapisz dokumenty i odśwież przeglądarkę
  Klikanie powoduje wstawianie niebieskich kropek
- (1) Ustaw **10 kropek**
  - najpierw wybierz w puste pole,
  - wybierz dwa pola zajęte **w odległości 5 kratek**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

(1)  lub  lub  itp.



9) Kreski

Kreski nie mogą nakładać się na siebie, ale mogą się przecinać. Z jednego punktu może wychodzić do 8 kresek. Każda komórka tablicy KRESKI zawiera liczbę, która opisuje kierunek wychodzenia kreski z tego pola. Każdy kierunek, to ustawiony bit. Kierunki z odpowiadającymi im bitami i współrzędnymi wektorów wektora pokazuje rysunek. Każdy bit w komórce tablicy KRESKI, to jedna kreska w konkretnym kierunku. 8 bitów tworzy liczby od 0 do 255 – to elementarz działania wszystkich cyfrowych urządzeń elektronicznych.



- Dokument **JS** - wklej nowe funkcje z ramki

```
function RysujKreske(k1,w1,k2,w2){
	var po=KWnaXY(k1,w1,BOK);
	var ko=KWnaXY(k2,w2,BOK);
	KR.beginPath();
	KR.lineWidth=4;
	KR.strokeStyle="black";
	KR.moveTo(po.x,po.y);
	KR.lineTo(ko.x,ko.y);
	KR.stroke();
}

function ZerujKreski(){
	for (var k=0;k<ILE;k++){
		KRESKI[k]=[];
		for (var w=0;w<ILE;w++)
			KRESKI[k][w]=0;
	}
}
function BITnaD(bit){
	var dk=0;
	var dw=0;
	if (bit==0) {dk=-1;dw=-1};
	if (bit==1) {dk=0;dw=-1};
	if (bit==2) {dk=1;dw=-1};
	if (bit==3) {dk=1;dw=0};
	if (bit==4) {dk=1;dw=1};
	if (bit==5) {dk=0;dw=1};
	if (bit==6) {dk=-1;dw=1};
	if (bit==7) {dk=-1;dw=0};
	return {dk,dw};
}
function CzyUstawionyBit(liczba,bit){
	return liczba & 1 << bit;
}
function RysujKreski(){
	for (var k=0;k<ILE;k++)
	for (var w=0;w<ILE;w++)
		if (KRESKI[k][w]>0){
			var kie=KRESKI[k][w];
			for (var b=0;b<8;b++){
				var bit=CzyUstawionyBit(kie,b);
				if (bit>0){
					var d=BITnaD(b);
					RysujKreske(k,w,k+d.dk,w+d.dw);
				}
			}
		}
}
```

Funkcja RYSUJKRESKE rysuje kreski od punktu do punktu
Funkcja ZerujKreski tworzy tablicę KRESKI, w której w komórkach zapamiętujemy położenia kresek wychodzących z kropki
Funkcja BITnaD zwraca współrzędne przesunięć w każdym kierunku: bit 0 – (-1,-1), bit 1 – (-1,0) itd.
Funkcja CzyUstawionyBit sprawdza liczbę, czy wybrany byt jest ustawiony
Funkcja RysujKreski przegląda tablicę i rysuje wszystkie kreski wychodzące z punktu

- Dokument **HTML** - wpisz instrukcje

```
for (var i=0;i<7;
MyszkaNapis();
```

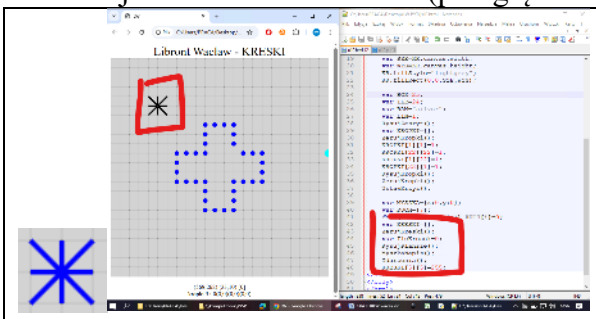
```
var KRESKI=[];
ZerujKreski();
var IleKresek=0;
```

Deklaracja tablicy na kreski i ustawianie początkowe tablicy
Deklaracja zmiennej, w której przechowujemy liczbę kresek

- Dokument **JS,** funkcja **RysujPlansze()**, wpisz
```
RysujKreski();
```
Najpierw rysujemy kreski (bądą pod spodem) a potem kropki

Dokument **HTML** - przestaw funkcję `RysujPlansze();` do
```
var IleKresek=0;
```
Aby narysowało planszę muszą być wcześniej zadeklarowane tablice KROPKI i KRESKI

- Dokument **HTML** - wpisz
```
KRESKI[5][5]=255;
```
W punkcie [5][5] ustawione wszystkie kreski
- Zapisz dokumenty i odśwież przeglądarkę
Na planszy powinien pojawić się kontrolny „krzyż" złożony z kresek
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

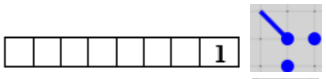


10) Kreski - bity

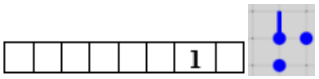
Każdy bit w komórce tablicy KRESKI, to jedna kreska w konkretnym kierunku. 8 bitów tworzy liczby od 0 do 255 – to elementarz działania wszystkich cyfrowych urządzeń elektronicznych.

0 – brak kresek

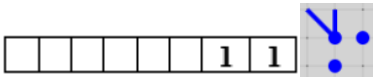
1 – kreska w kierunku (-1,-1) – ustawiony bit 0



2 – kreska w kierunku (0,-1) – ustawiony bit 1

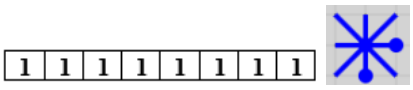


3 – kreska w kierunkach (-1,-1) i (0,-1) – ustawione bity 0 i 1



…

255 – wszystkie kreski



*Za pojedyncze kreski w każdym kierunku odpowiadają tzw. „wagi bitów": **1, 2, 4, 8, 16, 32, 64, 128***
Gdy poprawnie wybrano kropkę początkową i kropki końcowe, to można narysować 4 kreski pomiędzy pięcioma kropkami.

- Dokument **HTML** - wpisz znaczniki
```
<br>
kreski <label id=idKRES> </label>
```
- Dokument **HTML** - usuń polecenie `KRESKI[5][5]=255;`
- Dokument **JS,** funkcja **MyszkaNapis()** - wklej instrukcje

```
var kre=KRESKI[kw.k][kw.w];
var kr="";
for (var b=0;b<8;b++)
      if (CzyUstawionyBit(kre,b)>0)
            kr="1"+kr;
```

```
      else
            kr="0"+kr;
idKRES.innerHTML=IleKresek+" ["+kre+"] "+ kr
KR.fillStyle="black";
KR.font = "normal 40px Arial";
KR.fillText(IleKresek, 5, 570);
```

zawartość tablicy KRESKI na wskazanej komórce oraz kierunki-bity

- Dokument **JS** - wklej nowe funkcje z ramki

```
function UstawKreski(){
      var k0=RUCH[1];
      var w0=RUCH[2];
      var k1=RUCH[3];
      var w1=RUCH[4];
      var k2=RUCH[5];
      var w2=RUCH[6];
      var d=KWnaD(k1,w1,k2,w2);
      var waga=DnaWAGA(d.dk,d.dw);
      for (var i=0;i<4;i++){
            var ki=k1+d.dk*i;
            var wi=w1+d.dw*i;
            KRESKI[ki][wi]=KRESKI[ki][wi]+waga;
      }
      var waga=DnaWAGA(-d.dk,-d.dw);
      for (var i=1;i<5;i++){
            var ki=k1+d.dk*i;
            var wi=w1+d.dw*i;
            KRESKI[ki][wi]=KRESKI[ki][wi]+waga;
      }
}
function DnaWAGA(dk,dw){
      var waga=0;
      if (dk==-1 && dw==-1) waga=1;
      if (dk== 0 && dw==-1) waga=2;
      if (dk== 1 && dw==-1) waga=4;
      if (dk== 1 && dw== 0) waga=8;
      if (dk== 1 && dw== 1) waga=16;
      if (dk== 0 && dw== 1) waga=32;
      if (dk==-1 && dw== 1) waga=64;
      if (dk==-1 && dw== 0) waga=128;
      return waga;
}
function KWnaD(k1,w1,k2,w2){
      var dk=k2-k1;
      var dw=w2-w1;
      if ((dk==0 && dw!=0) || (dw==0 && dk!=0) || (Math.abs(dk)==Math.abs(dw))){
            if (dk!=0) dk=div(dk,Math.abs(dk));
            if (dw!=0) dw=div(dw,Math.abs(dw));
      }
      else{
            dk=0;
            dw=0;
      }
      return {dk,dw};
}
```

Funkcja UstawKreski:
- pobiera parametry ustawionych kropek z RUCH
- zamienia współrzędne kreski na wagę bitu kierunku
- ustawia bit wagi na czterech pierwszych kropkach
- ustawia przeciwny bit wagi na czterech ostatnich kropkach
Funkcja SnaWAGA zamienia kierunek kreski na wagę bitu
Funkcja KwnaD zamienia współrzędne początku i końca kreski na przesunięcia o 1 i sprawdza poprawność

- Dokument **JS,** funkcja **CzyPoprawnyRuch()** - wpisz

```
var OK5=true;
var waga=DnaWAGA(dk,dw);
```

Sprawdzamy, czy w tablicy KRESKI wybrana kropka nie ma już ustawionego bitu tego kierunku

- Dokument **JS,** funkcja **CzyPoprawnyRuch()** - wpisz

```
if ((i<4) && (KRESKI[ki][wi] & waga) > 0) OK5=false;
```

Sprawdzamy, czy w tablicy KRESKI wybrana kropka nie ma już ustawionego bitu tego kierunku
Ale bez ostatniej kropki, która już może łączyć się z inną kreską

- Dokument **JS,** funkcja **CzyPoprawnyRuch()**
  - dopisz na końcu nową zmienną

```
var OK=OK0 & OK1 & OK2 & OK3 & OK4 & OK5;
```

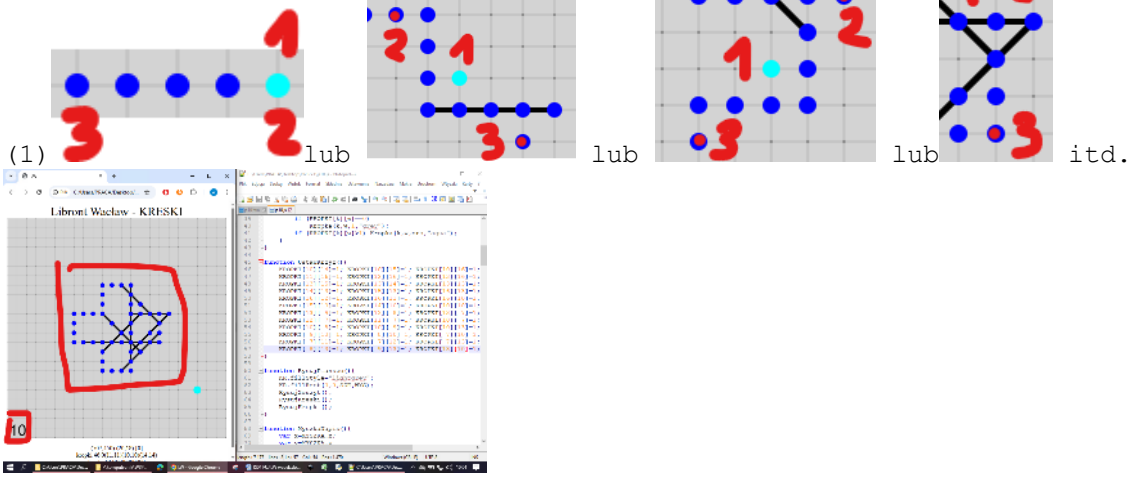
```
KROPKI[RUCH[1]][RUCH[2]]=1;
}
```

- Dokument **JS,** funkcja **Ruch()** - wpisz

```
UstawKreski();
IleKresek++;
```

Rysowanie kresek i zwiększenie licznika kresek

- Zapisz dokumenty i odśwież przeglądarkę
  Można ustawiać kreski, gdy jest poprawny ruch
- (1) Ustaw przynajmniej **10 poprawnych kresek**
  - najpierw wybierz w pustą kratkę
  - potem wybierz dwa pola zajęte **w odległości 5 kratek**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



11) Od nowa

```
    kreski <label id=idKRES> </label>
</center>
```

- Dokument **HTML** -

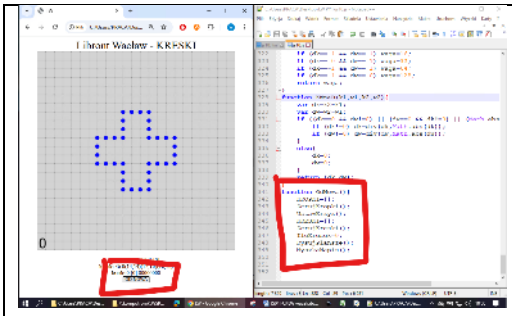
```
<br>
<input type=button value="OD NOWA" onclick=OdNowa()>
```

- Dokument **JS** - wklej tekst z ramki

```
function OdNowa(){
      KROPKI=[];
      ZerujKropki();
      UstawKrzyz();
      KRESKI=[];
      ZerujKreski();
      IleKresek=0;
      RysujPlansze();
      MyszkaNapis();
}
```

Funkcja zeruje tablice i zmienne – rozpoczynamy grę od nowa

- Zapisz dokumenty i odśwież przeglądarkę
  Przycisk OD NOWA resetuje wszystkie ustawione kreski
- Wstaw kilka kresek i wciśnij przycisk **OD NOWA**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



12) Cofnij ruch

- Dokument **HTML** – wpisz definicję przycisku

```
<input type=button value="COFNIJ" onclick=CofnijRuch()>
```

- Dokument **HTML** - wpisz deklarację zmiennej tablicowej

```
var RUCHY=[];
```

- Dokument **JS,** funkcja **Ruch()** - wklej polecenia z ramki

```
RUCHY[IleKresek]=[];
for (var i=0;i<7;i++)
      RUCHY[IleKresek][i]=RUCH[i];
```

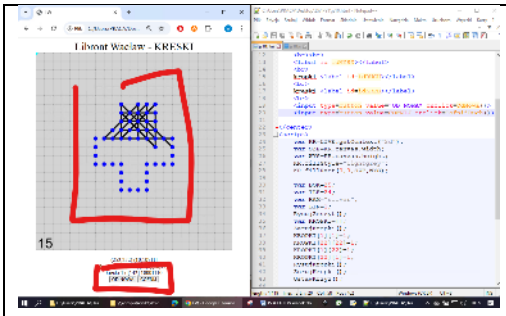
Jeżeli poprawny ruch, to współrzędne trzech kropek zapamiętujemy w tablicy RUCHY

- Dokument **JS** - wklej nowe funkcje z ramki

```
function CofnijRuch(){
      if (IleKresek<=0) return false;
      var k0=RUCHY[IleKresek][1];
      var w0=RUCHY[IleKresek][2];
      var k1=RUCHY[IleKresek][3];
      var w1=RUCHY[IleKresek][4];
      var k2=RUCHY[IleKresek][5];
      var w2=RUCHY[IleKresek][6];
      KROPKI[k0][w0]=0;
      var d=KWnaD(k1,w1,k2,w2);
      var waga=DnaWAGA(d.dk,d.dw);
      for (var i=0;i<4;i++){
            var ki=k1+d.dk*i;
            var wi=w1+d.dw*i;
            KRESKI[ki][wi]=KRESKI[ki][wi] - waga;
      }
      var waga=DnaWAGA(-d.dk,-d.dw);
      for (var i=1;i<5;i++){
            var ki=k1+d.dk*i;
            var wi=w1+d.dw*i;
            KRESKI[ki][wi]=KRESKI[ki][wi] - waga;
      }
      RUCHY[IleKresek]=[];
      IleKresek--;
      KR.canvas.click();
      RUCH[0]=0;
}
```

Funkcja CofnijRuch
- pobiera parametry ostatniego ruchu z tablicy RUCHY
- zeruje kropkę
- odejmuje wagę z czterech pierwszych kropek
- odejmuje odwrotną wagę z czterech ostatnich kropek
- symuluje kliknięcie w planszę - przerysowanie

- Zapisz dokumenty i odśwież przeglądarkę
  Przycisk COFNIJ powoduje anulowanie ostatniego ruchu
- Wstaw kilka kresek, a następnie cofnij
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



13) Gramy

- Wstaw poprawnie co najmniej **20 kresek**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)