



1.1 Informatika obecně

Co si představíš pod pojmem informatika?

Co to informatika podle tebe je?

Kde se s informatikou dneska můžeš setkat?

Kde se znalosti z informatiky využívají?

Co zkoumají vědci zabývající se informatikou?



Léto teenagerů 2025
(14. až 18. července)

Informatika trochu jinak

Jaká informatická témata by tě zajímala?
O čem by ses chtěl dozvědět více?

1.

2.

3.

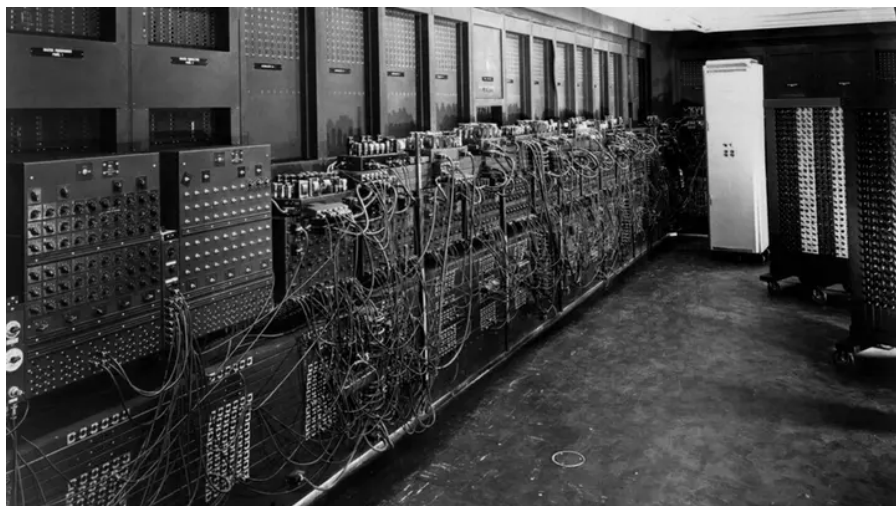
4.

Jak nás informatika ovlivňuje?
V čem všem nás ovlivňuje?

1.2 Úvod k počítačům

Počítač je **stroj**, který plní nějaké námi vymyšlené **úkoly** (něco spočítej, zobraz něco na obrazovce, ...), které zapíšeme jako **příkazy**.

První **elektronické počítače** tak jak je známe dnes vznikali v druhé polovině 20. století nicméně jakési představy počítače se objevovali i v předchozích stoletích, tehdy ale lidé ještě neznali **elektronické součástky** jako jsou diody, tranzistory a elektronky. Nebojte, nemusíme přesně rozumět co tyhle slova znamenají, jednoduše technologie nebyly ještě tak daleko aby lidé dokázali sestavit počítače tak, jak se známe dnes. Snažili se o sestavení počítačů mechanických (třeba i s parním pohonem).



Obrázek 1: Sálavý počítač ENIAC

Elektronické počítače začínali jako obrovské stroje zabírající plochu celých pater domů, tzv. **sálavé počítače** vznikali od poloviny 20. století. Postupným **zlepšováním elektronických součástek**, zejména jejich **zmenšováním**, nás dovedlo až do stavu, kdy je celý počítač velký jako **placička** co se nám vleze do kapsy (a zároveň je tahle placička nesrovnatelně výkonnější než sálavé počítače minulosti).

Základními součástmi dnešních počítačů jsou **procesor**, **základní deska**, **grafická karta**, **operační paměť**, **trvalá paměť** a **zdroj**. Důležité jsou i zařízení, která do počítače připojujeme (např. pomocí USB-A nebo HDMI konektoru) a která rozšiřují funkčnost počítače (např. obrazovka, klávesnice), těmto zařízením říkáme **vstupní a výstupní zařízení**.



K čemu nám ale vlastně počítače jsou?
Jakou plní funkci?

Z jakých (základních) částí se počítače skládají?

Název komponenty	K čemu slouží?
1.	
2.	
3.	
4.	
5.	
6.	



1.3 Programovací jazyky

Každý počítač **ovládáme příkazy**. Existují **různé úrovně složitosti** a **různé sady** těchto příkazů. Platí, že složitější příkazy jsou **přeloženy** na ty jednodušší a těm nejjednodušším říkáme **instrukce**. Jsou to **elementární úkony procesoru** jako je sečtení dvou čísel nebo uložení výsledku do paměti počítače.

Programovací jazyky pracují s příkazy, které jsou **složitější**, jsou tedy překládány na ty jednodušší (na instrukce). Programovací jazyky slouží k psaní **kódu programu**, který má plnit nějaký úkol. **Různé programovací jazyky** se hodí k **různým úkolům**. Některé se používají k programování aplikací pro Windows (C#), jiné zase pro aplikace na telefonu (Swift, Kotlin) či pro webové aplikace (JavaScript, TypeScript, Python). **Liší se svými funkcemi** a proto **nepoužíváme jeden univerzální** jazyk na vše. Obecně si jsou ale programovací jazyky celkem **podobné** a proto, když se naučíme jeden bude pro nás **jednodušší** se naučit další pro jiné úkoly.

```
n = int(input("Zadejte číslo: "))
total_sum = 0
for i in range(1, n + 1):
    total_sum += i
print(f"Součet čísel od 1 do {n} je {total_sum}")
```

Obrázek č.2: Ukázka Python kódu

My si během týdne ukážeme programování v Pythonu, co je dnes velmi používaný programovací jazyk, zejména při vývoji webových aplikací nebo při zpracování dat.



1.4 Instrukce (příkazy)

Instrukce na ovládání člověka:	
Název instrukce	Popis instrukce
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	
11.	
12.	
13.	
14.	
15.	



2.1 Binární soustava

Číselné soustavy používáme pro reprezentaci nějakého počtu, množství něčeho, jednoduše pro reprezentaci čísel. Znáš už **desítkovou soustavu**, bereme ji jako základní soustavu pro počítání čehokoli, počítače to mají ale jinak, používají pro **počítání a ukládání dat soustavu binární neboli dvojkovou**. Desítková používá **10 číslic** (0 až 9), binární ale **pouze 2** (0 a 1). Pro nás informatiky je důležité rozumět **převodům** mezi desítkovou a binární soustavou, jelikož pokud máme něco uložené v paměti počítače, tak to jsou pro nás pouze jedničky a nuly, potřebujeme si to tedy **převést do desítkové soustavy** aby nám to **dávalo smysl** (popřípadě přeložíme uložené jedničky a nuly podle nějakého kódování na písmena, o tom ale později).

Jedna jednička nebo nula se nazývá **bit**. V paměti jsou informace uloženy po blocích, typicky po 8 bitech a tomuto bloku říkáme **bajt**. Tedy jeden bajt může vypadat například: 1010 1101 nebo 1101 0100. Všimni si že pro lepší čitelnost píšeme tyto bloky **s mezerou uprostřed**.

Zkus přijít na to, proč počítače používají binární soustavu?

Jak si počítače mezi sebou posílají informace (data)?



Každý jeden bit v čísle zapsaném v binární soustavě má svou **hodnotu** v **desítkové soustavě**, díky tomu můžeme celkem jednoduše mezi těmito soustavami **převádět**, prostě jen sečteme **hodnoty bitů na pozicích kde je jednička** (z binární do desítkové) nebo **umístíme jedničky** na pozice binárního čísla tak, aby nám dávalo nějaké v desítkové soustavě (z desítkové do binární).

Tabulka hodnot pozic:

Číslo pozice	8.	7.	6.	5.	4.	3.	2.	1.
Hodnota pozice v desítkové soustavě	128	64	32	16	8	4	2	1
Mocnina dvojky	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Číslo v binární soustavě	1	1	1	1	1	1	1	1

Příklady převodů z binární do desítkové:

Číslo v binární soustavě	Výpočet převodu podle hodnot pozic	Číslo v desítkové soustavě
0000 0011	$2+1$	3
0010 0010	$32+2$	34
0011 1101	$32+16+8+4+1$	61

Příklady na procvičení převodů z binární do desítkové:

Číslo v binární soustavě	Výpočet převodu podle hodnot pozic	Číslo v desítkové soustavě
0000 0100		
0001 0001		
0000 1100		
0001 0110		
0010 1000		
1001 0001		
0110 0100		
0111 0110		



Příklady převodů z desítkové do binární:

Číslo v desítkové soustavě	Výpočet převodu podle hodnot pozic	Číslo v binární soustavě
15	$8+4+2+1$	0000 1111
23	$16+4+2+1$	0001 0111
132	$128+4$	1000 0100
0	0	0000 0000

Příklady na procvičení převodů z desítkové do binární:

Číslo v desítkové soustavě	Výpočet převodu podle hodnot pozic	Číslo v binární soustavě
8		
12		
17		
23		
41		
65		
133		
138		
260		

2.2 Počítání v binární soustavě

Počítače počítají v binární soustavě, nás bude zajímat **hlavně sčítání**, jak to ale dělají? Funguje to podobně jako **sčítání pod sebou** v desítkové soustavě (což znáte asi tak ze třetí třídy). **Sčítáme tedy postupně jednotlivé cifry** a speciálně pokud tedy sčítáme dvě jedničky dostaneme nulu a jedna jde dál, sčítání nuly a jedničky je jednička a sčítání tří jedniček (jedna z předchozího sčítání), pak je jednička a jedna jde dál. Lépe to lze ale pochopit na příkladech níže (můžete si pak výsledek i sčítance převést do desítkové soustavy a sečtené číslo v desítkové soustavě a výsledek by mělo být **stejně číslo**, zkuste to pro ověření).



Příklady:

$$\begin{array}{r} 1) \quad 0100 \ 0101 \\ + 0011 \ 0100 \\ \hline 0111 \ 1001 \end{array}$$

$$\begin{array}{r} 2) \quad 1001 \ 0000 \\ + 0010 \ 1111 \\ \hline 1011 \ 1111 \end{array}$$

$$\begin{array}{r} 3) \quad 1100 \ 1010 \\ + 1001 \ 0010 \\ \hline 1 \ 0101 \ 1100 \end{array}$$

$$\begin{array}{r} 4) \quad 0000 \ 1101 \\ + 1100 \ 1101 \\ \hline 1101 \ 1010 \end{array}$$

Příklady k procvičení sčítání:

$$\begin{array}{r} 1) \quad 0000 \ 1100 \\ + 0000 \ 0000 \\ \hline \end{array}$$

$$\begin{array}{r} 2) \quad 0100 \ 0001 \\ + 0010 \ 1001 \\ \hline \end{array}$$

$$\begin{array}{r} 3) \quad 0011 \ 0010 \\ + 0001 \ 0010 \\ \hline \end{array}$$

$$\begin{array}{r} 4) \quad 0001 \ 0001 \\ + 1101 \ 0111 \\ \hline \end{array}$$

$$\begin{array}{r} 5) \quad 0100 \ 1100 \\ + 0110 \ 1100 \\ \hline \end{array}$$

$$\begin{array}{r} 6) \quad 0111 \ 1111 \\ + 0110 \ 1101 \\ \hline \end{array}$$

$$\begin{array}{r} 7) \quad 0111 \ 0010 \\ + 0011 \ 0110 \\ \hline \end{array}$$

$$\begin{array}{r} 8) \quad 0101 \ 1111 \\ + 0101 \ 0111 \\ \hline \end{array}$$

Prostor pro mezi výpočty a poznámky:



Zatím jsme pracovali jen s čísly kladnými a nulou (tedy nezápornými), v počítači často chceme reprezentovat i **čísla záporná** (např. -3, -12, ...), to děláme pomocí tzv. **dvojkového doplňku**. Ten vypadá tak, že kladnému číslu k číslu které chceme mít záporné (tedy číslo opačné) nejdříve **invertujeme** všechny bity (jinak negujeme nebo obrátíme) tak, že **z jedničky uděláme nulu a opačně** a následně k výsledku **přičteme** jednu **jedničku**. Důležité je vědět, že díky dvojkovému doplňku pořád **funguje sčítání** čísel (tedy můžeme sečíst např. 8 a -3) a tedy umíme už i **odečítat** jelikož $8 + (-3) = 8 - 3$.

Příklady převodu do dvojkového doplňku (na záporné číslo):

Záporné číslo v desítkové soustavě	Opačné číslo k tomuto číslu v binární soustavě	Invertované číslo v binární soustavě	Číslo s přičtenou jedničkou (už dvojkový doplněk)
-4	0000 0100	1111 1011	1111 1100
-7	0000 0111	1111 1000	1111 1001

Příklady k procvičení převodu do dvojkového doplňku:

Záporné číslo v desítkové soustavě	Opačné číslo k tomuto číslu v binární soustavě	Invertované číslo v binární soustavě	Číslo s přičtenou jedničkou
-2			
-10			
-12			
-6			
-10			

Prostor pro mezivýpočty a poznámky:



Ted' si to vše **spojíme dohromady**, čísla v závorkách nejdříve **převeďte** do binární soustavy a následně **spočítejte** v binární soustavě příklad (pozn. Odčítání provedeme tak, že číslo které odečítáme převedeme na číslo záporné tedy pomocí dvojkového doplňku, při použití dvojkového doplňku pracujeme s nějakou danou přesností, tj. délkou čísla, cokoli co se dostane mimo tuto délku zahazujeme, my budeme pracovat s délkou čísla 8 bitů, 8 cifer čísla), výsledek převeďte do desítkové soustavy a zkontrolujte, že vám to vyšlo správně:

1)	(1+4)	2)	(4+5)	3)	(7+12)	4)	(8-1)	5)	(12-4)
	+		+		+		+		+

6)	(32+12)	7)	(28+24)	8)	(18-8)	4)	(68+25)	10)	(59-9)
	+		+		+		+		+

Prostor pro mezivýpočty a poznámky:



2.3 Kódování písmen v počítači

Kódování (nebo jinak i reprezentace) **písmen** a tedy i **textu** v počítači je vyřešeno **překládovou tabulkou**, která **obsahuje číselný kód** pro **písmena** (a, b, c, ...) a **speciální znaky** (@, \$, !, ...). Tedy každé číslo v této tabulce znamená nějaké **písmeno či znak**. Slova a věty jsou tedy representována čísly (s kterými už umíme pracovat a hlavně je převádět z desítkové do binární soustavy). Historicky vzniklo **několik takovýchto překládových tabulek**, nejúspěšnější a **nejpoužívanější** je ale **ASCII tabulka** a její následná rozšíření (např. UTF-8). My budeme pracovat se základním kódováním ASCII tabulky, kde jeden znak je 8 bitů, tedy jeden bajt, ta má ale **svá omezení** (např. **neobsahuje písmena s háčky ani čárkami**).

ASCII Tabulka

0	NUL	16	DLE	32	SPC	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	\$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	
13	CR	29	GS	45	-	61	=	77	M	93]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

Tedy například slovo „Ahoj“ je v počítači zapsáno jako čísla 65, 104, 111 a 106 (samozřejmě v binární soustavě).



2.4 Šifrovačka s ASCII tabulkou

Vaším úkolem je vyluštit zakódované slova, jména a slovní spojení, později si o nich popovídáme. Mezivýpočty si pište kdekoli kde je volné místo.

1)

0100 1000 *****	0110 0101 *****	0110 1100 *****	0110 1100 *****	0110 1111 *****
0101 0111 *****	0110 1111 *****	0111 0010 *****	0110 1100 *****	0110 0100 *****

2)

0100 1100 *****	0110 1001 *****	0110 1110 *****	0111 0101 *****	0111 0011 *****			
0101 0100 *****	0110 1111 *****	0111 0010 *****	0111 0110 *****	0110 0001 *****	0110 1100 *****	0110 0100 *****	0111 0011 *****

3)

0101 0011 *****	0111 0100 *****	0110 0101 *****	0111 0110 *****	0110 0101 *****
0100 1010 *****	0110 1111 *****	0110 0010 *****	0111 0011 *****	

4)

0100 1100 *****	0110 1001 *****	0110 1110 *****	0111 0101 *****	0111 1000 *****
--------------------	--------------------	--------------------	--------------------	--------------------



3.1 Počítačové sítě

Jeden počítač je sice dost užitečný, když ale počítače **propojíme** tak, aby mohly **komunikovat** mezi sebou ať už přes **drát** a tedy elektrickými signály (např. Ethernet) nebo **bezdrátově** a tedy pomocí elektromagnetického záření (např. Wi-Fi), můžeme je udělat **ještě užitečnější**. Počítače začínali jako **samostatné jednotky**, komunikace mezi nimi se ale ukázala jako užitečná a tak začaly vznikat první **počítačové sítě** (60. léta minulého století). Počítačovou síť bychom tedy mohli definovat (=popsat) jako **propojení několika zařízení** aby mohla probíhat jejich vzájemná komunikace.








Počítačové sítě mohou být **malé**, třeba jen síť pro malou firmu, kde je **připojeno pár počítačů**, nebo mohou být **obrovské**. Největší počítačovou sítí je síť označovaná jako **Internet**, kam jsou připojeny **miliardy zařízení**, která mohou spolu komunikovat. Internet je **spojení** obrovského množství **menších sítí**.

Kde se můžeme setkat s počítačovými sítěmi?
Jaká zařízení připojuješ do počítačové sítě?

K čemu ti počítačové sítě (hlavně Internet) slouží?
Co přes ně posíláme?

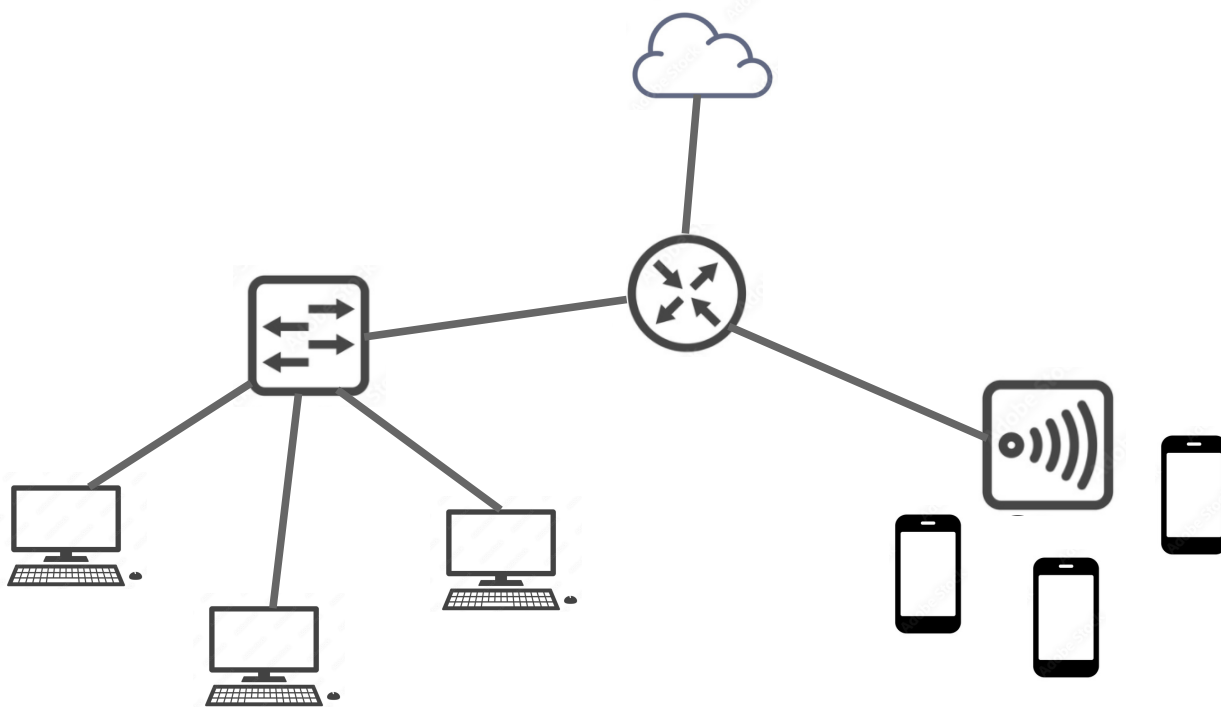
3.2 Zařízení potřebná pro chod počítačové sítě

K tomu, aby počítačová síť fungovala je **potřeba několik zařízení**, která se nám **starají o spojení**. Uvedeme si jak je budeme **značit obrázky** a k čemu slouží, uvedeme si i jak budeme značit počítač či celou síť. Obrázky zařízení se používají při **návrhu sítě** (třeba když je postaven nový dům, tak je potřeba navrhnout počítačovou síť v něm).

Obrázek:			
Název zařízení:	Stolní počítač	Telefon	Označení celé sítě nebo internetu
Vysvětlení zařízení:	Počítačové sítě propojují počítače, potřebujeme značit samotný počítač a to jednoduše monitorem a klávesnicí	Do počítačové sítě můžeme připojit i telefony, ty budeme značit touto ikonou	Ikonou obláčku budeme značit nějakou síť či podsít nebo celý internet. (anglicky je cloud)
			
Směrovač (Router)	Síťový přepínač (switch)	Wi-Fi anténa	Propojení drátem
Šipky v kolečku znamenají různá propojení sítí, směrovač je zařízení, které propojuje více počítačových sítí, typicky dvě	Šipky podobně jako u směrovače znamenají různá propojení, přepínač je ale zařízení, které propojuje ostatní zařízení jen v rámci jedné počítačové sítě	Částmi kruhu vycházejícími z jednoho bodu značíme Wi-Fi anténu, ta propojuje zařízení bezdrátově, síťový přepínač (switch) naopak drátově	Propojení drátem budeme označovat jednoduše čarou, zařízení, která jsou propojená drátem spojíme tedy jednoduše čarou

3.3 Ukázka počítačové sítě

Počítačová síť (její schéma, obrázek) může vypadat například takto:



Popište síť na obrázku, jaké a kolik má zařízení?
Kam je připojena? Kde by mohla být taková síť?



Léto teenagerů 2025
(14. až 18. července)

Informatika trochu jinak

3.4 Vaše domácí počítačová síť

Nakresli počítačovou síť, kterou máte doma, kolik má zařízení? Jsou tam připojeny telefony, počítače, notebooky (notebook značte označením pro počítač), nebo máte televizi připojenou k internetu, je to drátová síť nebo bezdrátová, nebo oboje?



3.5 Jak se počítače poznají?

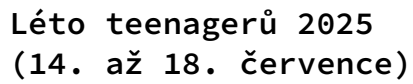
Už víme jaká zařízení síť obsahuje, jak ale funguje komunikace mezi nimi a hlavně jak se mezi sebou počítače **poznají, kdo je kdo?** K tomu slouží **adresy** (IP adresy, internet protocol adresy). Adresa je podobně jako adresa vašeho domu **jednoznačný identifikátor**, tedy podle adresy poznáme konkrétní zařízení. IP adresa (konkrétně verze 4, existuje i verze 6) je jednoduše **číslo**, samozřejmě v uložené v počítači v binární soustavě skládající se z **32 bitů** (32 cifer) a tedy máme k dispozici 4294967296 (něco přes 4 miliardy) různých adres.

Když počítač chce poslat zprávu jinému počítači, musí uvést **jeho adresu**, musí si ji zjistit. IP adresu svého zařízení (ať už počítače nebo telefonu) můžete **najít v nastavení sítě**, když vám nějaký jiný počítač posílá zprávu, píše ji právě na tuto adresu.

IP adresy píšeme typicky jako 4 čísla oddělená tečkami (8 bitů je vždy jedno tohle číslo, $8 \times 4 = 32$ bitová adresa), při návrhu sítě píšeme typicky k zařízením i jejich IP adresy, které jim až budeme síť fyzicky stavět přiřadíme. Tedy IP adresa může vypadat například 172.50.32.2 nebo (vyluštěte):

- | | | | | |
|----|-----------|-----------|-----------|-----------|
| 1) | 1100 0000 | 1010 1000 | 0000 0000 | 0000 0001 |
| | . | . | . | . |
| 2) | 0111 1111 | 0000 0000 | 00000000 | 0000 0001 |
| | . | . | . | . |
| 3) | 0000 1000 | 0000 0100 | 0000 0110 | 0000 0101 |
| | . | . | . | . |
| 4) | 1010 1100 | 0001 0000 | 1111 1110 | 0000 0001 |
| | . | . | . | . |

Samotné rozpoznávání kde je který počítač v obrovské síti jako je internet je **dost složité**, hlavní je vědět, že se **směrovače** (routery) **orientují podle adres**, hledají síť kde je **adresa počítače**, kam posíláme zprávu a až tuhle síť postupně najdou tak už „zaměří“ přímo počítač.



Kolik existuje cest od počítače k serveru?

Místo pro mezivýpočty:



4.1 Programovací jazyk Python

(Opakování:) **Programovací jazyky** jsou sady možných **příkazů**, které mají určitá **pravidla** jak tyhle příkazy psát (=syntaxe jazyka). Příkazy, které napíšeme v nějakém programovacím jazyku jsou **převezeny na instrukce** pro počítač a následně tyhle **instrukce ovládají počítač** (říkáme, že počítač vykonává instrukce).

Na SPŠE Olomouc se budeme věnovat programovacímu jazyku **Python**. Což je jazyk **vysoké úrovně** (je **jednodušší** a **přehlednější** pro práci než jazyky nižší úrovně). Python se používá pro vývoj **webových aplikací**, pro automatizaci a jednoduché **scripty** (=kratší programy) pro ulehčení práce, či pro **analýzu dat** nebo pro práci s **umělou inteligencí**.

Jak ale příkazy v Pythonu **vypadají**? Ukážeme si níže, doporučuji se ale pro **více ukázek a více popsané příkazy**, mrknout na moje **materiály ke kroužku** Programovací v Pythonu (který probíhal v DDM Olomouc) na adrese: https://github.com/askoldH/python_krouzek .

Důležitým konceptem programovacích jazyků jsou **datové typy**. Existují datové typy jako jsou **celá čísla** (integers, int), **desetinná čísla** (floating point numbers, float), **texty** (strings, str) či **boolean hodnoty** (bool) neboli hodnoty true (1, platí, pravda) nebo false (0, nepravda, neplatí). S různými datovými typy můžeme dělat **různé operace**. Čísla **sčítat** (a+n), **odčítat** (a-b), **násobit** (a*b), **dělit** (a/b), **porovnávat** (a<b, a>b, a<=b, a>=b, a==b). **Texty porovnávat** (text_1==text_2) nebo také třeba **sčítat** (text_1+text_2). Boolean hodnoty (true, false) pak používáme u **podmínek** (ukážeme si níže).

Dalším důležitým konceptem jsou **proměnné**. **Proměnná** je **místo** („šuplíček“) v **paměti** s názvem (jménem proměnné), tedy když například pracujeme s načtenými daty od uživatele, uložíme si je do proměnné.

Pojďme ale už na některé příkazy:

Zápis příkazu	Význam příkazu
jmeno_promenno = něco	Přiřazení do proměnné (může to být číslo, text, ...)
print(něco)	Vytiskne na obrazovku to, co vložíme do závorky, př. print("Ahoj") vytiskne Ahoj.
input()	Načte text od uživatele, dovoluje uživateli interagovat s počítačem, často chceme ukládat načtená data do proměnných, tedy například nactena_data = input()
a < b	Porovnání dvou čísel, pokud je rovnost platná, výsledek je true, pokud je neplatná, pak je false, tohle zapsání je tzn. podmínka



<code>a + b</code>	Sečtení dvou čísel
<code>if podmínka: blok_kódu</code>	Pokud platí podmínka (její hodnota je true) pak se provede blok kódu (poznámka: jako blok kódu bereme jakékoli další příkazy)
<code>if podmínka_1: blok_kodu_1 elif podmínka_2: blok_kodu_2 elif podmínka_3: blok_kodu_3 else: blok_kodu_4</code>	Podmínky mohou být i více rozvětvené. V takové případě se provede první blok kódu, kterého podmínka platí, nebo se provede poslední else větev
<code>while podmínka: blok_kodu</code>	Dokud podmínka platí, blok kódu se vykonává pořád dokola
<code>for i in range(0, 5): blok_kodu</code>	Blok kódu se provede 5x, můžeme přeložit jako pro i v rozsahu 0 až 5 (bez pětky) se prováděj
<code>seznam = [neco, neco]</code>	Seznam proměnných, seznam může obsahovat čísla, slova, ...
<code>seznam.append(neco)</code>	Přidání do prvku seznamu
<code>seznam[2]</code>	Zacílení hodnoty v seznamu na 3. pozici (jelikož číslujeme prvky od nuly, tedy 0., 1. a 2.)
<code>def jmeno_fce(): blok_kodu</code>	Vytvoření nové funkce, nastavujeme si název funkce, funkce je v podstatě pojmenovaný blok kódu
<code>jmeno_fce()</code>	Vyvolání funkce, tohle volání spustí kód funkce

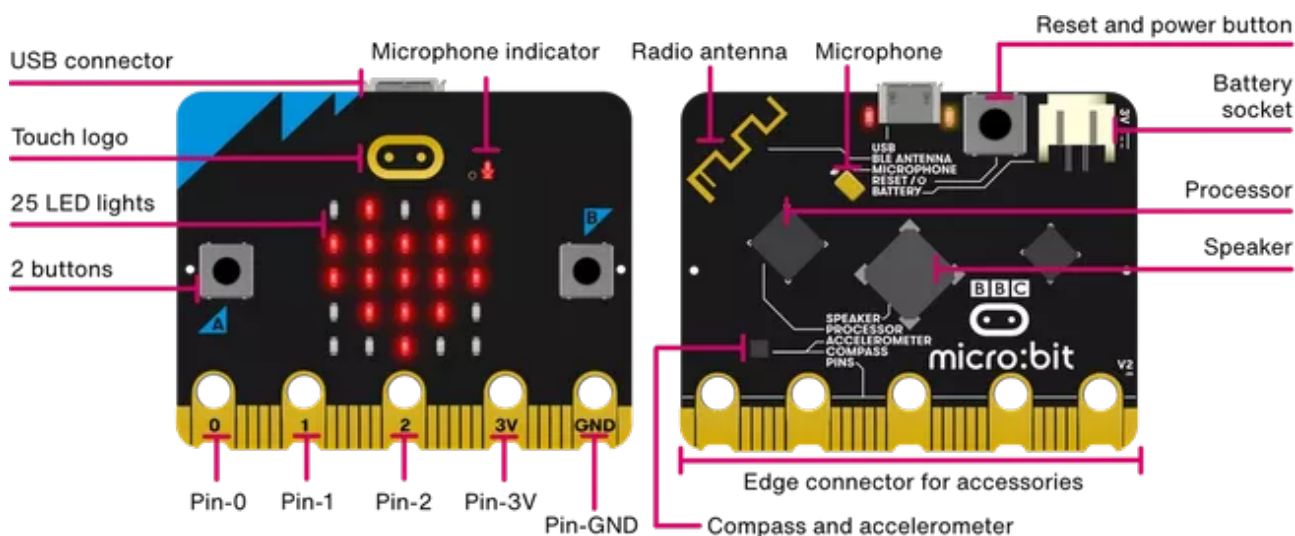
Existuje **mnohem více příkazů**, tohle jsou ale ty **základní**, které budeme používat (budeme používat i jejich **odvozeniny**). Pro více popsání příkazů se podívejte na poznámky ke kroužku zmíněné výše. Z těchto příkazů budeme skládat první programy v Pythonu, budeme pracovat v prostředí VSCode, které je dnes velmi oblíbené kvůli své rozšířitelnosti a jednoduchosti, pokud by jste chtěli programovat ale nevíte jako prostředí si vybrat, VSCode určitě nebude chyba.

Příklady na procvičení programování v Pythonu budeme čerpat ze stránek kroužku, tedy: https://github.com/AskoldH/python_krouzek/tree/main/ukoly . Pro ty, kteří jste pokročilejší si mám vymyšlené složitější úkoly.

4.2 Micro:bit

Micro:bit je **malý počítač**, který dokáže **ovládat** led diody či reproduktor, taky dokáže kontrolovat, jestli někdo nezmáčknul tlačítka. Micro:bit má ještě o něco více funkcí nicméně nám pro naši výstavku budou stačit tyto.

Obrázek **schématu Micro:bitu**, neboli **kde co najdeme** (vše si ale vysvětlíme a i ukážeme):



Micro:bit se programuje buď pomocí blokových příkazů nebo Python kódu, programujte v tom, co vám přijde příjemnější, hlavní je výsledek!

Prostředí Micro:bitu je velmi intuitivní a budeme si s ním hrát. U Micro:bitu je nejlepší si sám zjistit co umí.

4.3 Výstavka

Jako naši výstavku uděláme společný projekt na Micro:bitech. Každý jeden Micro:bit (který naprogramuje každý z vás) bude plnit roli v celé výstavce skládající se z 11 Micro:bitů.



Léto teenagerů 2025
(14. až 18. července)

Informatika trochu jinak

5.1 Hledání informatiky

Název zařízení:	Popis zařízení:



Hodnocení příměšťáku

Pojďme mi dát zpětnou vazbu, hodnotte slovy, vždy k otázce napište pár slov, pomůže to s vytvořením lepšího programu na příště.

Myslíš, že ses toho během tábora naučil dost?	
Jak tě bavil program tábora?	
Působil jsem na tebe jako vedoucí (Aski) dobře?	
Věděl jsem toho jako vedoucí dost?	
Která aktivita tě bavila nejvíce? Která nejméně?	
Vycházel jsi s ostatními kolegy během tábora?	
Myslíš si, že sis našel během tohoto týdne nové kamarády?	
Chutnaly ti obědy?	
Chutnaly ti svačinky?	
Vyhovoval ti systém pracovních listů?	
Chtěl bys jít na podobný příměšťák i příští rok?	
Doplň cokoli dalšího, ať už o programu, jídle, sportovních aktivitách, informatických aktivitách, o mě (vedoucím) či o prostorech kde jsme byli, chybělo ti na příměšťáku něco? ...	



Léto teenagerů 2025
(14. až 18. července)

Informatika trochu jinak
