12. lekce

Převody mezi dvojkovou a šestnáctkovou soustavou

Miroslav Jílek

Převody mezi dvojkovou a šestnáctkovou číselnou soustavou

Základní myšlenkou je : 2^4 = 16

To znamená, že čtyři číslice dvojkové číselné soustavy tvoří jednu číslici šestnáctkové soustavy.

Převod provádíme tak, že od desetinné tečky (čárky) v obou směrech (doleva – celočíselná část, doprava – desetinná část) vytváříme skupiny po čtyřech znacích dvojkové soustavy. Pokud by v poslední skupině byly méně než čtyři číslice, pak tuto skupinu doplníme nulami na čtyři číslice - vždy ve směru od desetinné tečky (čárky):

Příklad:
Převeďte číslo zapsané ve dvojkové číselné soustavě do šestnáctkové soustavy:

1011011001.11101

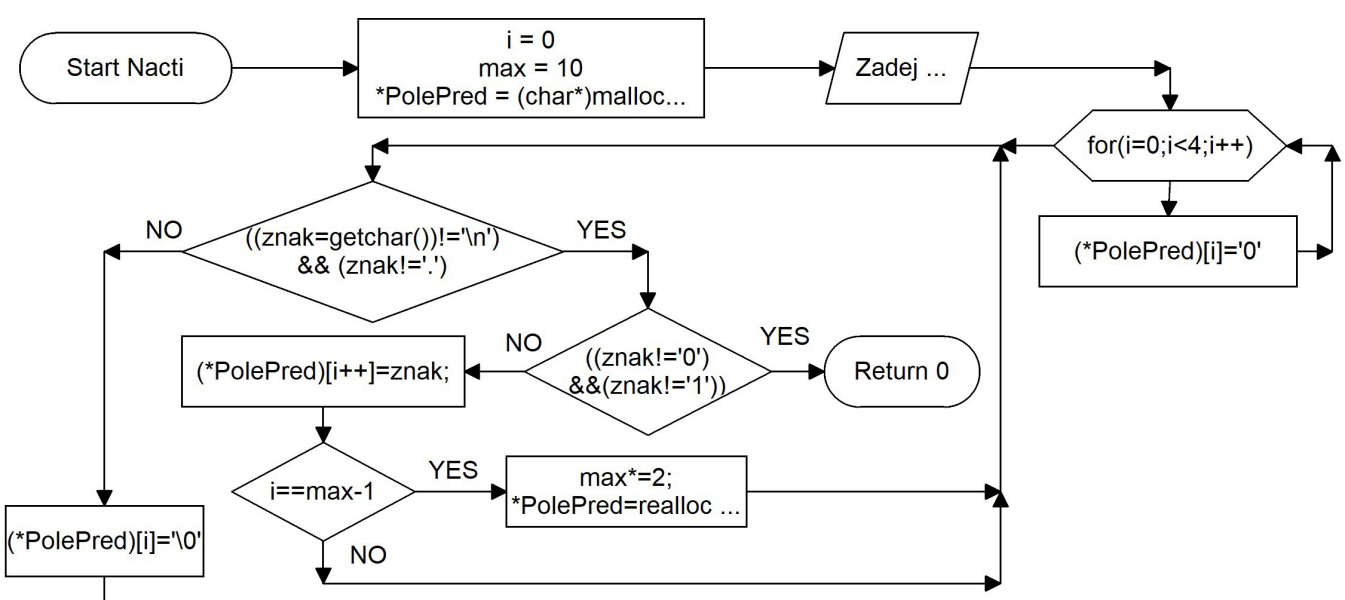
doplníme nulami na skupiny po čtyřech:

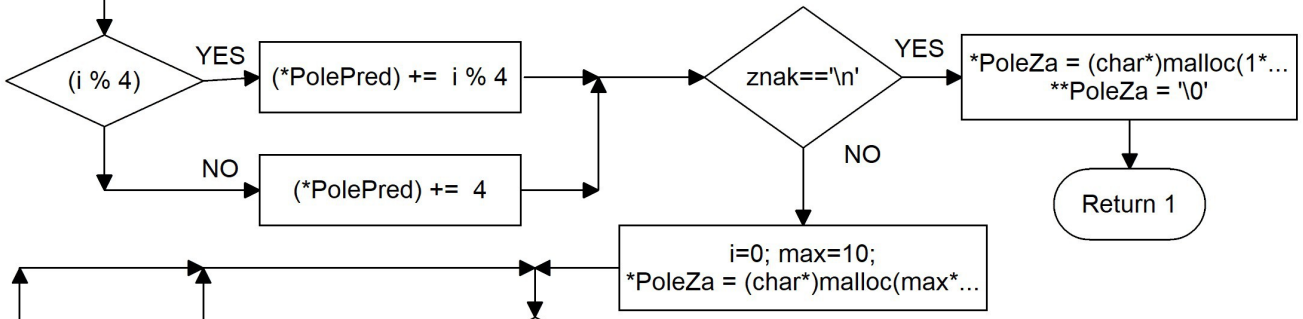
0010 1101 1001.1110 1000

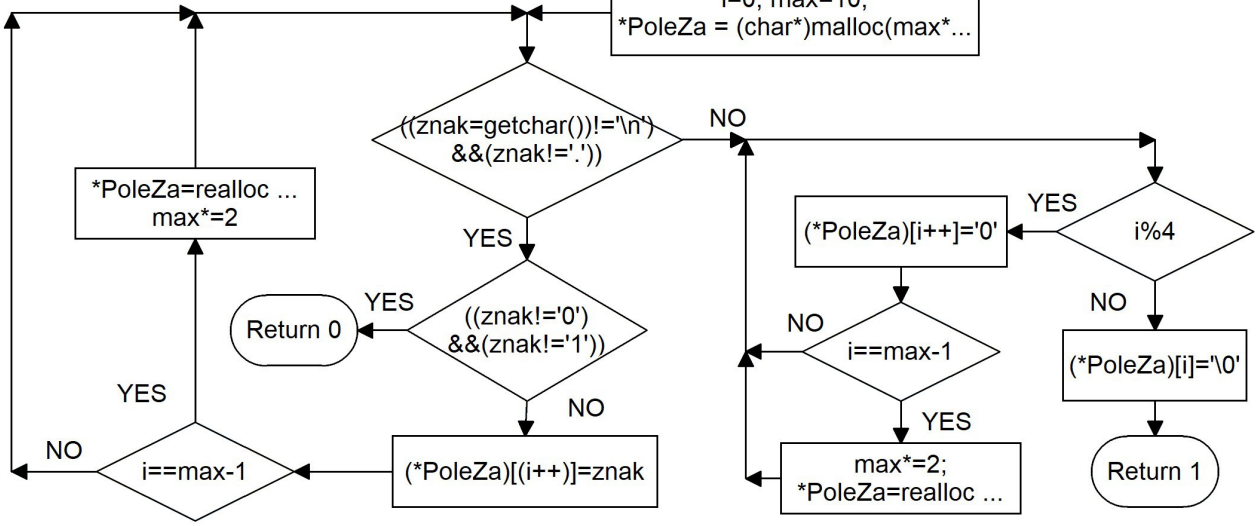
každou čtyřčlennou skupinu nahradíme jednou číslicí šestnáctkové soustavy

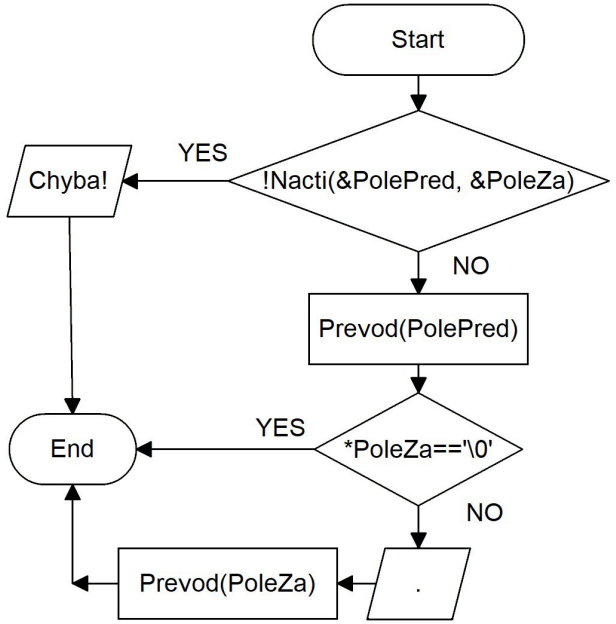
0010	**1101**	**1001**	**.**	**1110**	**1000**
2	**D**	**9**	**.**	**E**	**8**

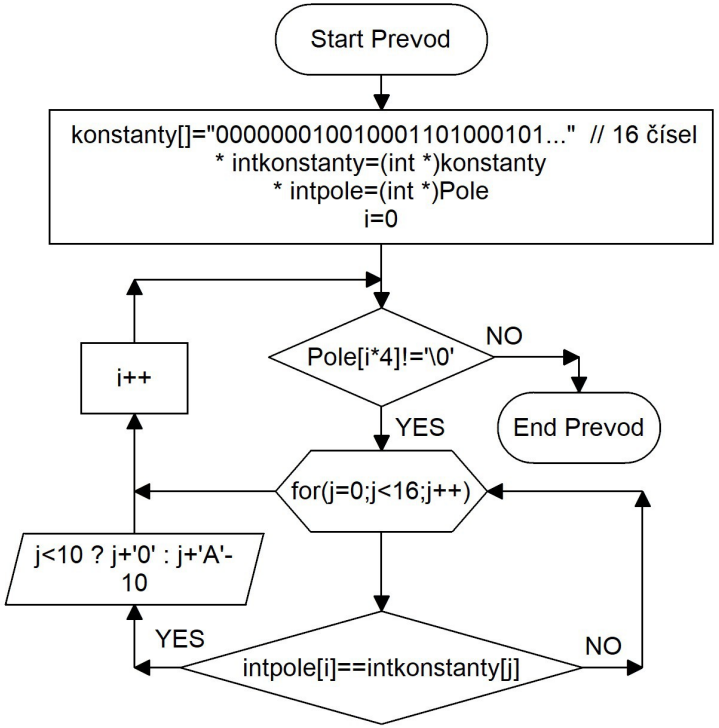
$$1011011001.11101_{(2)} = 2D9.E8_{(16)}$$


Start Nacti
i = 0
max = 10
PolePred = (char)malloc...
Zadej ...
for(i=0;i<4;i++)
(*PolePred)[i]='0'
((znak=getchar())!='\n')
&& (znak!='.')
NO
YES
(*PolePred)[i++]=znak;
((znak!='0')
&&(znak!='1'))
NO
YES
Return 0
i==max-1
YES
max*=2;
*PolePred=realloc ...
NO
(*PolePred)[i]='\0'


(i % 4)
YES
(*PolePred) += i % 4
NO
(*PolePred) += 4
znak=='\n'
YES
PoleZa = (char)malloc(1*...
**PoleZa = '\0'
NO
Return 1
i=0; max=10;
PoleZa = (char)malloc(max*...


((znak=getchar())!='\n')
&&(znak!='.'))
NO
YES
Return 0
YES
((znak!='0')
&&(znak!='1'))
NO
(*PoleZa)[(i++)]=znak
YES
NO
i==max-1
*PoleZa=realloc ...
max*=2
(*PoleZa)[i++]='0'
YES
i%4
NO
NO
i==max-1
YES
max*=2;
*PoleZa=realloc ...
(*PoleZa)[i]='\0'
Return 1


Start
!Nacti(&PolePred, &PoleZa)
YES
Chyba!
NO
Prevod(PolePred)
*PoleZa=='\0'
YES
End
NO
Prevod(PoleZa)
.


Start Prevod
konstanty[]="00000001001000110100010l..." // 16 čísel
* intkonstanty=(int *)konstanty
* intpole=(int *)Pole
i=0
Pole[i*4]!='\0'
NO
YES
End Prevod
for(j=0;j<16;j++)
i++
j<10 ? j+'0' : j+'A'-
10
YES
intpole[i]==intkonstanty[j]
NO

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int Nacti(char **PolePred, char **PoleZa ) //ukazatel na ukazatel na char
{
	int i=0, max=10;
	char znak;
	*PolePred = (char*)malloc(max*sizeof(char));
	printf("Zadej cislo v dvojkove soustave (desetinna tecka): ");
	for(i=0;i<4;i++) (*PolePred)[i]='0'; //doplneni nul na zacatek pole
	while(((znak=getchar())!='\n')&&(znak!='.'))
	{
		if ((znak!='0')&&(znak!='1')) return 0;
		(*PolePred)[(i++)]=znak;
		if (i==max-1)
		{
			max*=2;
			*PolePred=realloc(*PolePred,max*sizeof(char));
		}
	}
	(*PolePred)[i]='\0';
	if (i % 4) (*PolePred) += i % 4; else (*PolePred) += 4;		//jestli
je i%4 tak posun adresu PolePred o i%4, jinak o 4
	if (znak=='\n')
	{
		*PoleZa = (char*)malloc(1*sizeof(char));
		**PoleZa = '\0'; //do pole o jednom znaku vložime symbol
		'\0' - konec pole
		return 1;
	}
	i=0;
	max=10;
	*PoleZa = (char*)malloc(max*sizeof(char));
	while(((znak=getchar())!='\n')&&(znak!='.'))
	{
		if ((znak!='0')&&(znak!='1')) return 0;
		(*PoleZa)[(i++)]=znak;
		if (i==max-1)
		{
			max*=2;
			*PoleZa=realloc(*PoleZa,max*sizeof(char));
		}
	}
	while(i%4)	// doplneni poctu znaků na nasobek 4
	{
		(*PoleZa)[i++]='0';
		if (i==max-1)
		{
			max*=2;
			*PoleZa=realloc(*PoleZa,max*sizeof(char));
		}
	}
	(*PoleZa)[i]='\0';
	return 1;
}
```

```
void Preved(char *Pole)
{
	const char konstanty[]="0000000100100011010001010110011111000100110101011110011011110111111";
	//všechny kombinace 4 znaku 1 a 0 (po ctyrech znacich), pro vytvoreni pole intkonstanty
	int * intkonstanty=(int *)konstanty; //pretypuje pole konstanty a vlozi ho do pole intkonstanty - bude mít 16 indexů - cisel - ze 4 charů po 8 bitech uděla 1 int po 32 bitech (fyzicky zůstane, zmeni se interpretace bitu)
	int * intpole=(int *)Pole; //stejne jako predchozi radek
	int i=0,j;
	while(Pole[i*4]!='\0')
	{
		for(j=0;j<16;j++)
			if (intpole[i]==intkonstanty[j])
			{
				printf("%c",j<10 ? j+'0' : j+'A'-10);
				break;
			}

		i++;
	}
}

int main (void)
{
	char * PolePred, *PoleZa;
	if (!Nacti(&PolePred, &PoleZa)) // posilame adresu adresy
	{
		printf("Chyba!\n");
		return 1;
	}
	Preved(PolePred);
	if (*PoleZa=='\0') return 0;
	printf(".");
	Preved(PoleZa);
	return 0;
}
```