Zad. 7.1

Napisz program `stack` umieszczający na stosie kolejno dwie zmienne `x` i `y` typu `int` z wartościami 1 i 2 oraz odczytaj adresy tych zmiennych. Skopiuj poniższy schemat do komentarza w programie i wypełnij go odpowiednimi wartościami.

```
&var1 [ ][ ][ ][ ]   var1
&var2 [ ][ ][ ][ ]   var2
```

Zad. 7.2 *

W programie `data` w sekcji danych zainicjowanych zadeklaruj zmienne `z1, z2, z3, z4` z wartościami od 1 do 4, a w sekcji danych niezainicjowanych zmienne `n1, n2, n3, n4` wszystkie typu `int`. Zmienne zadeklaruj w następującej kolejności:

```
z1 z2  n1 n2  z3 n3  n4 z4
```

Skopiuj poniższy schemat do komentarza w programie oraz wypełnij go odpowiednimi wartościami.

```
heap  [ ][ ][ ][ ]   heap

&var1 [ ][ ][ ][ ]   var1
&var2 [ ][ ][ ][ ]   var2
&var3 [ ][ ][ ][ ]   var3
&var4 [ ][ ][ ][ ]   var4

&var5 [ ][ ][ ][ ]   var5
&var6 [ ][ ][ ][ ]   var6
&var7 [ ][ ][ ][ ]   var7
&var8 [ ][ ][ ][ ]   var8

main  [ ][ ][ ][ ]   text
```

Zad. 7.3

W programie `heap` zarezerwuj na stercie dwa bloki pamięci o rozmiarze typu `int`. Adresy tych bloków zapisz w zmiennych odpowiednio `x` i `y`. Wypisz te adresy oraz oblicz rozmiar pierwszego bloku pamięci. Pod adresem `x` i `y` umieść odpowiednio liczby 1 i 2 typu `int`, a następnie je wypisz. Skopiuj poniższy schemat do komentarza w programie i wypełnij go odpowiednimi wartościami.

```
     [ ][ ][ ][ ]
     [ ][ ][ ][ ]
     [ ][ ][ ][ ]
var1 [ ][ ][ ][ ]   *(int*)var1

     [ ][ ][ ][ ]
     [ ][ ][ ][ ]
     [ ][ ][ ][ ]
var2 [ ][ ][ ][ ]   *(int*)var2
```

Czy do wskaźnika typu `void*` można stosować operator wyłuskania?

Ile razy należy wywołać funkcję `malloc`, aby obliczyć rozmiar przydzielonego obszaru pamięci?

Jaki najmniejszy obszar pamięci przydziela na stercie system operacyjny dla kodu 32 i 64 bitowego?

Zad. 7.4

W pliku `konwersje.txt` dokonaj konwersji liczb:

- dziesiętne 11 na liczbę binarną
- dziesiętne 99 na liczbę binarną *
- szesnastkowe 10AF na liczbę binarną
- szesnastkowe 3A58 na liczbę binarną *

Zad. 7.5

Napisz program `konwersje` implementujący następujące funkcje:

```
int dec2bin(int x);  // na liczbie dziesiętnej
int dec2bin2(int x); // na liczbie ósemkowej *
int bin2dec(int x);  // *
void dec2byte(unsigned int x); // reprezentacja little-endian *
```

- jaką maksymalną liczbę binarną można zapisać na liczbie dziesiętnej typu `int`?

- jaką maksymalną liczbę binarną można zapisać na liczbie ósemkowej typu `int`? *

- jaka jest wartość dziesiętna maksymalnej liczby binarnej zapisanej na liczbie dziesiętnej typu `int`?

- jaka jest wartość dziesiętna maksymalnej liczby binarnej zapisanej na liczbie ósemkowej typu `int`? *

- dla jakich wartości parametrów aktualnych powyższe funkcje będą działać poprawnie?

Przykładowa sesja:

```
dec2bin(1023) = 1111111111

bin2dec(1111111111) = 1023

dec2byte(1023) = [255][003][000][000]
```

Zad. 7.6

Napisz program `number` wyliczający wartość liczby bez znaku pod adresem `p` na podstawie jej `n` bajtowej reprezentacji little-endian przy pomocy funkcji:

- funkcja `polinomial` wylicza wartość wielomianu w sposób klasyczny

```
int polinomial(unsigned char *p, int n);
```

- funkcja `horner` wylicza wartość wielomianu schematem Hornera *

```
int horner(unsigned char *p, int n);
```

Przetestuj funkcje dla reprezentacji jedno, dwu i czterobajtowych. Reprezentację liczby należy określić przy pomocy tablicy typu `char`. Wskazówka:

```
https://balois.pl/java/proste/wielomiany.htm
```

Przykładowa sesja dla reprezentacji `[1][2][3][4]`:

```
number(0065FEC4, 4) = 67305985
number(0065FEC4, 4) = 67305985
```

- pod jakim adresem znajduje się reprezentacja liczby 67305985?

Zad. 7.7 *

W domu zainstaluj środowisko do programowania w języku asemblera zgodnie instrukcją umieszczono w na stronie z materiałami.