**演習問題１**

#include <stdio.h>

/\*

\* main関数です.

\*/

int main(void){

char c;

short int s;

int i;

long int l;

float f;

double d;

/\* 変数のアドレスを表示します \*/

printf("char型変数のアドレス: %p\n", &c);

printf("short␣int型変数のアドレス: %p\n", &s);

printf("int型変数のアドレス: %p\n", &i);

printf("long␣int型変数のアドレス: %p\n", &l);

printf("float型変数のアドレス: %p\n", &f);

printf("double型変数のアドレス: %p\n", &d);

/\* 正常終了します \*/

return 0;

}

**実行結果**

char型変数のアドレス: 000000E8161FFD3F

short␣int型変数のアドレス: 000000E8161FFD3C

int型変数のアドレス: 000000E8161FFD38

long␣int型変数のアドレス: 000000E8161FFD34

float型変数のアドレス: 000000E8161FFD30

double型変数のアドレス: 000000E8161FFD28

**演習問題２**

#include <stdio.h>

/\*

\* main関数です.

\*/

int main(void){

/\* カウンタ変数 \*/

int i;

/\* 通常の変数 \*/

int a = 10;

/\* 配列 \*/

int b[] = {1, 2, 3};

/\* ポインタ変数 \*/

int \*p;

/\* ポインタ変数を初期化します \*/

p = &a;

/\* 参照しているデータの値を表示します \*/

printf("a = %d\n", \*p);

/\* 配列の要素を参照して表示します \*/

for (i = 0; i < 3; i++) {

/\* 配列の要素を参照します \*/

p = &b[i];

/\* 参照しているデータの値を表示します \*/

printf("b[%d] = %d\n", i, \*p);

}

/\* 正常終了します \*/

return 0;

}

**実行結果**

a = 10

b[0] = 1

b[1] = 2

b[2] = 3

**演習問題3**

#include <stdio.h>

/\*

\* 整数の和と差を計算します.

\*/

void add\_and\_sub(int, int, int \*, int \*);

/\*

\* main関数です.

\*/

int main(void){

/\* 整数 \*/

int x = 10, y = 5;

/\* 和と差 \*/

int add, sub;

/\* 整数の和と差を計算します \*/

add\_and\_sub(x, y, &add, &sub);

/\* 整数の和と差を表示します \*/

printf("%d + %d = %d\n", x, y, add);

printf("%d - %d = %d\n", x, y, sub);

/\* 正常終了します \*/

return 0;

}

/\*

\* 整数の和と差を計算します.

\*/

void add\_and\_sub(int a, int b, int \*add, int \*sub){

\*add = a + b;

\*sub = a - b;

}

**実行結果**

10 + 5 = 15

10 - 5 = 5

**演習問題４**

#include <stdio.h>

/\*

\* 配列の各要素の和を計算します.

\*/

void add\_arrays(int \*, int \*, int \*, int);

/\*

\* main関数です.

\*/

int main(void){

/\* カウンタ変数 \*/

int i;

/\* 整数配列 \*/

int a[4] = {1, 3, 5, 7};

int b[4] = {2, 4, 6, 8};

int c[4];

/\* 配列の各要素の和を計算します \*/

add\_arrays(a, b, c, 4);

/\* 配列の各要素の和を表示します \*/

printf("c = [ ");

for (i = 0; i < 4; i++) {

printf("%d ", c[i]);

}

printf("]\n");

/\* 正常終了します \*/

return 0;

}

/\*

\* 配列の各要素の和を計算します.

\*/

void add\_arrays(int \*a, int \*b, int \*c, int n){

/\* カウンタ変数 \*/

int i;

/\* 各要素の和を計算します \*/

for (i = 0; i < n; i++) {

\*(c + i) = \*(a + i) + \*(b + i);

// c[i] = a[i] + b[i];

}

}

**実行結果**

c = [ 3 7 11 15 ]

]

**演習問題５**

#include <stdio.h>

/\*

\* 小文字と大文字を反転します.

\*/

void convert\_lower\_and\_upper(char \*);

/\*

\* main関数です.

\*/

int main(void){

/\* 文字配列 \*/

char string[100];

/\* 文字列をキー入力します \*/

printf("文字列を入力: ");

scanf("%s", string);

/\* 変換前の文字列を表示します \*/

printf("変換前の文字列: %s\n", string);

/\* 小文字と大文字を反転します \*/

convert\_lower\_and\_upper(string);

/\* 変換後の文字列を表示します \*/

printf("変換後の文字列: %s\n", string);

/\* 正常終了します \*/

return 0;

}

/\*

\* 小文字と大文字を反転します.

\*/

void convert\_lower\_and\_upper(char \*string)

{

/\* 文字配列へのポインタ \*/

char \*p;

/\* ポインタを初期化します \*/

p = string;

/\* ヌル文字が見つかるまで反転を繰り返します \*/

while (\*p != '\0') {

if (\*p >= 'a' && \*p <= 'z') {

/\* 大文字へ変換します \*/

\*p = \*p - 'a' + 'A';

} else if (\*p >= 'A' && \*p <= 'Z') {

/\* 小文字へ変換します \*/

\*p = \*p - 'A' + 'a';

} else {

/\* 何もしません \*/

}

/\* ポインタを1文字分増やします \*/

p++;

}

}

**実行結果**

文字列を入力: wINDOWSxp

変換前の文字列: wINDOWSxp

変換後の文字列: WindowsXP