Imię i nazwisko _____ **Numer indeksu** _____

Wypełnij drukowanymi literami *Czas pisania: 75 minut, data:* 6 styczeń 2019

Uwaga: we wszystkich programach należy założyć, że dołączone są biblioteki iostream, stdlib oraz dostępna jest przestrzeń nazw std. Sprawdzaniu podlegają jedynie miejsca wyznaczone na odpowiedź. W przypadku stwierdzenia błędu lub niejednoznaczności w pytaniu, należy czytelnie napisać komentarz wyjaśniający napotkany problem. Test oceniany jest w skali 0-100 pkt (próg zaliczenia = 50%).

Zad. 1. (20 pkt. = 7*2 + 6 pkt.) Wyróżnione pola uzupełnij, tak aby program poprawnie się kompilował oraz wypisywał na wyjście:

AB

Podając odpowiedź należy wpisać we wszystkich polach BŁĄD jeśli rozwiązanie nie istnieje.

Uwaga: każde pole jest oceniane na 2 pkt. Wypisanie na wyjście właściwego tekstu – 6 pkt.

Uwaga: zakładamy, że wywołanie funkcji *malloc* się powiodło. Zauważ, że program nie zwalnia pamięci, co nie wpływa na sposób interpretacji kodu (zwalnianie pamięci pomijamy ze względu na ograniczone miejsce na teście).

```
typedef struct A {
    char c;
    struct A *next;
} list_t;
list_t *add( list_t *h, _____________ ) {
    list_t *n = (list_t *)malloc( sizeof(list_t) );

    _____________ = _____________;
    n->c = c;
    return n;
}
int main() {
    list_t *l = NULL;

    _____________ = add( _____________, 'A' );

    _____________ = add( _____________, 'B' );
    while ( l != NULL ) {
        cout << l->c;
        l = l->next;
    }
    return 0;}
```

Zad. 2. (20 pkt. = 4*5 pkt.) Podaj zawartość wskazanych elementów tablicy *a* bezpośrednio przed zakończeniem realizacji funkcji *main*.

Odpowiedź:

a[0] = _____

a[1] = _____

a[2] = _____

a[3] = _____

```
void m( int *a, int *b, int *c, int n ){
    int i;
    for (i=0; i < n; i++ ) {
        if ( b[i] % 3 > c[i] % 3 )
            a[i] = b[i];
        else
            a[i] += c[i];
    }
}
int main() {
    int a[] = { 1,2,0,2, 2,1,0,1, 2,0,2,1 };
    m( a+8, a+4, a, 4 );
    m( a+4, a+8, a, 4 );
    m( a, a+4, a+8, 4 );
    return 0;
}
```

Zad. 3. (20 pkt. = 2*10 pkt.) Podaj co zostanie wypisane na ekran w poszczególnych wywołaniach *cout*.

Pierwsze *cout*: _____

Drugie *cout*: _____

```
int count( int *a, int n) {
    if ( n > 1 )
        return count( a+1, n-1 ) + ( a[0]>a[1] ? 1:0 );
    else
        return 0;
}
int main() {
    int x[] = { 3,5,8,4,6,3,3,9,6,3,7,6,1 };
    cout << count( x, (sizeof x) / (sizeof x[1]) );
    cout << count( &(x[2]), 7 );
    return 0;
}
```

Zad. 4. (20 pkt. = 4*5 pkt.)
Wyróżnione pola uzupełnij, tak aby program poprawnie się kompilował, gwarantował brak błędów wykonania oraz w wyniku wykonania wypisał na ekran liczbę *7*. Podając odpowiedź:

- nie należy używać znaków: ;,[]
- należy wpisać BŁĄD jeśli rozwiązanie nie istnieje
- należy wpisać BRAK jeśli pole powinno pozostać puste.

```
int g( _______________ ) {

    _____________;
}
int f( _______________ ) {

    g( _____________ );
    return x;
}
int main() {
    cout <<  f( 5 );
    return 0;
}
```

Zad. 5. (20 pkt. = 5*4 pkt.)
Podaj tekst, który zostanie wypisany na wyjściu w wyniku wykonania poszczególnych instrukcji *cout* (w miejsce na odpowiedź oznaczonym etykietą "Instrukcja x" wpisz tekst wypisany przez instrukcję *cout* z komentarzem "/* I-x */"). Wpisz *ERR* jeśli nie można jednoznacznie stwierdzić co zostanie wypisane na ekran. Kodowanie liczb w systemie binarnym przyjmujemy tak jak omówiono na wykładzie, tzn. U2. Jeśli jakaś instrukcja powoduje zapis poza tablicą lub innego rodzaju błąd wykonania, to w odpowiedzi wpisz *ERR* i kontynuuj realizację programu z pominięciem tej instrukcji.

Odpowiedzi:

Instrukcja 1: _____

Instrukcja 2: _____

Instrukcja 3: _____

Instrukcja 4: _____

Instrukcja 5: _____

```
#define W  u+h
int main() {
    int *p = (int *) malloc( 100*sizeof(int) );
    int h=5, u=4, *v = &(p[30]), y[] = {4,3,2,1,0};

    cout << (12 ^ 3); /* I-1 */
    v += (h << 2);
    cout << v; /* I-2 */
    cout << (v-p)/sizeof(int); /* I-3 */
    cout << ( W*W ); /* I-4 */
    cout << &(y[3]) - &(*y); /* I-5 */
    return 0;
}
```