

ISO/ANSI C – zakończenie

Zmienne systemowe (PATH, LIB, itd.) można odczytać używając funkcji `getenv`

```
char *getenv( const char *varname );
```

Przykład:

```
char *pathvar;  
pathvar = getenv( "PATH" );  
if( pathvar != NULL )  
    printf( "PATH variable is: %s\n", pathvar );
```

© UKSW, WMP, SNS, Warszawa

209

209

ISO/ANSI C – zakończenie

Polecenie systemowe można wywołać używając funkcji `system`

```
int system( const char *command );
```

Przykład:

```
system( "type crt.txt" );  
/* wypisuje w oknie konsoli zawartość pliku  
   tekstowego crt.txt */
```

© UKSW, WMP, SNS, Warszawa

210

210

ISO/ANSI C – zakończenie

Aby zakończyć działanie programu musi być wykonana instrukcja: `exit`, `abort`, lub `return` w funkcji `main`

```
void abort( void );
```

Przerywa działanie programu, nie zwraca sterowania do procesu nadrzędnego ale wyświetla komunikat:

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.

© UKSW, WMP, SNS, Warszawa

211

211

ISO/ANSI C – zakończenie

```
void exit( int status );
```

Funkcja kończy działanie programu w trybie prawidłowego zakończenia

Standardowo przyjmuje się dwie wartości status:

0 (`EXIT_SUCCESS`) – program zakończony prawidłowo;

1 (`EXIT_FAILURE`) - program zakończony niepoprawnie

Przykład:

```
exit( 0 );
```

© UKSW, WMP, SNS, Warszawa

212

212

ISO/ANSI C – zakończenie

Funkcja `exit` przez wyjście z programu wywołuje wszystkie funkcje wskazane wywołaniem funkcji `atexit`. Może być ich do 32.

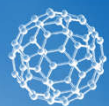
Przykład:

```
void fn1(void) {  
    printf( "do jutra.\n" );  
}  
void fn2(void) {  
    printf( "Do widzenia " );  
}  
int main( void ) {  
    atexit( fn1 );  
    atexit( fn2 );  
    exit( 0 );  
} /* Co pojawi się w oknie konsoli? */
```

© UKSW, WMP, SNS, Warszawa

213

213



Koniec części wykładu dotyczącej ISO/ANSI C

214