Wstęp do programowania w języku C

Lista zadań 2 (ver. 1)

1. (10/5) Napisać program, który zamienia w tekście ze standardowego wejścia polskie litery w standardzie WINDOWS-1250 na UTF-8 lub odwrotnie (w zależności od parametru wywołania programu: win2utf lub utf2win). Przetworzony tekst należy wypisać na standardowym wyjściu. Dodatkowo, jeśli po kropce i ewentualnych spacjach występuje mała litera (także polska), to należy ją zastąpić odpowiednią dużą. Jeśli jesteś początkującym programistą, to możesz zastąpić kodowanie UTF-8 przez ISO-8859-2. Do czytania i pisania pojedynczych bajtów znaków użyć odpowiednio funkcji `getchar` i `putchar`.

   Uwaga: Aby program mógł czytać argumenty wywołania programu, funkcja *main* musi mieć nagłówek: `int main(int argc, char *argv[])`, a w jej treści należy sprawdzić, czy argumenty wywołania programu zostały podane:

   ```
   if (argc <= 1) { printf("Brak argumentow \n"); return 1; }
   ```

   Jeśli argument został podany, to jego wartość można sprawdzić instrukcją

   ```
   if (strcmp(argv[1],"iso2utf") == 0) ...
   ```

   gdzie funkcja `strcmp` jest w standardowej bibliotece (trzeba dołączyć <string.h>) i służy do porównywania napisów.

2. (10/10) Napisać program, który przepisze tekst ze standardowego wejścia na standardowe wyjście umieszczając na wyjściu słowa rozdzielone pojedynczą spacją w zadanym przedziale kolumn $m \ldots n$, gdzie $m < n < 200$. Oznacza to, że każdy wiersz powinien się zaczynać $m-1$ znakami spacji, po których występuje ciąg kolejnych słów (co najmniej jedno) oddzielonych pojedynczą spacją, a ostatni znak ostatniego słowa w wierszu znajduje się pozycji $\leq n$, gdzie pozycje (kolumny) w wierszu numerujemy od 1. Wartości m i n powinny być argumentami wywołania programu. Jeśli w tekście wejściowym pojawi się słowo dłuższe niż $n - m + 1$, to można przerwać działanie programu z czytelnym komunikatem. Słowem nazywamy tu niepusty ciąg znaków nie zawierający tzw. białych znaków, czyli spacji, tabulacji czy znaku nowej linii.

   Uwaga: Aby program mógł czytać argumenty wywołania programu, funkcja *main* musi mieć nagłówek: `int main(int argc, char *argv[])`, a w jej treści należy sprawdzić, czy argumenty wywołania programu zostały podane:

   ```
   if (argc <= 1) { printf("Brak argumentow \n"); return 1; }
   ```

   a następnie przeczytać te argumenty instrukcjami:

   ```
   sscanf(argv[1], "%d", &m); sscanf(argv[2], "%d", &n);
   ```

   Argumenty wywołania programu można podać w CodeBlock-sie wybierając z menu: *Project -> Set programs' arguments...* Trzeba mieć zdefiniowany projekt w CodeBlock-sie dla takiego programu.

3. (10/10) Napisać program, który rozwiązuje zadanie opisane w Moodlu jako *Lista 2 zadanie 3*. Rozwiązanie tego zadania będzie sprawdzane automatycznie przez sprawdzarkę Moodlową.

25 października 2021 *Marek Piotrów*