PRZETWARZANIE TEKSTU

ZADANIA

Z1. Uruchom i przeanalizuj poniższy program.

```c
#include <stdio.h>
int main(void) {
  int t;
  char last;
  while ((t = getchar()) != EOF)
    last = t;
  putchar(last);
  putchar('\n');
}
```

Z2. Uruchom i przeanalizuj poniższy program.

```python
import sys
for line in sys.stdin:
    for ch in line:
        last= ch
print(last)
```

Z3. Uruchom i przeanalizuj poniższy program.

```python
import sys
for line in sys.stdin:
    for x in line:
      if 'A' <= x and x <= 'Z':
           ch= ord(x) + ord('a') - ord('A')
           print(chr(ch), end='')
      else:
           print(x, end='')
```

Z4. Uruchom i przeanalizuj poniższy program.

```c
int main(void) {
  char x, ch;
  while (scanf("%c", &x) != EOF) {
    if ('A' <= x && x <= 'Z') {
      ch = x + ('a' - 'A');
      printf("%c", ch);
    }
    else
      printf("%c", x);
  }
}
```

Z5. Uruchom i przeanalizuj poniższy program.

```c
#include <stdio.h>
int main(void) {
       int c, Num = 0;
       while ((c = getchar()) != EOF)
              if (c == 'a') Num++;
       printf("%d\n", Num);
       return 0;
}
```

Z6. Uruchom i przeanalizuj poniższy program.

```c
#include <stdio.h>
int main(void) {
	int c;
	while ((c = getchar()) != EOF)
		if ('A' <= c && c <= 'Z')
			putchar(c + 'a' - 'A');
		else putchar(c);
	return 0;
}
```

Z7. Uruchom i przeanalizuj poniższy program.

```c
#include <stdio.h>
int main(void) {
	char txt[100];
	int Num = 0;
	while (scanf("%s", &txt) != EOF)
		Num++;
	printf("%d\n", Num);
	return 0;
}
```

Z8. Uruchom i przeanalizuj poniższy program.

```python
Expected= "Warsaw"
Provided= input()
if Expected == Provided:
    print("OK")
else:
    print("Wrong!")
```

Z9. Uruchom i przeanalizuj poniższy program.

```c
#include <string.h>
#include <stdio.h>
int main() {
	char Expected[] = "Warsaw";
	char Provided[100];
	scanf("%s", Provided);
	if (!strcmp(Expected, Provided))
		printf("OK\n");
	else printf("Wrong!\n");
	return 0;
}
```

Z10. Uruchom i przeanalizuj poniższy program.

```c
#include <string.h>
#include <stdio.h>
int main(void) {
	char Expected[100], Provided[100];
	strcpy(Expected, "Warsaw");
	scanf("%s", &Provided);
	if (!strcmp(Expected, Provided))
		printf("OK\n");
	else printf("Wrong!\n");
	return 0;
}
```

Z11. Uruchom i przeanalizuj poniższy program.

```
#include <stdio.h>
#include <string.h>
int main(void) {
       char Text[] = "Go";
       printf("%d\n", strlen(Text));
       printf("%d\n", sizeof Text);
       return 0;
}
```

Z12. Uruchom i przeanalizuj poniższy program.

```
import sys
for line in sys.stdin:
    x= line.split()
    if x[3] == "I1":
       print(x[1], x[0])

dla następującego wejścia:

Jerzy Nawrocki   43089  I1
Jane  Kowalski   43780  I2
Adam  Malinowski 43990  I1
```

Z13. Uruchom i przeanalizuj poniższy program.

```
import sys
total=0
NR= 0
for line in sys.stdin:
    NR+= 1
    x= line.split()
    NF= len(x)
    total+= NF
print("Fields: " + str(total))
print("Rows: " + str(NR))

dla następującego wejścia:

If you
have
a hammer, everything
looks like a nail
```

Z14. Uruchom i przeanalizuj poniższy program.

```
import sys
import re
for line in sys.stdin:
    if re.search("ne", line):
       print(line, end='')

dla następującego wejścia:

It’s a favourite project of mine,
A new value of Pi to assign.
I  would fix it at 3,
For it's simpler, you see,
Than 3 point 1 4 1 5 9
```

Z15. Uruchom i przeanalizuj poniższy program.

```
#include <stdio.h>
#include <stdlib.h>
#include <regex.h>
#define maxSize 100
int main(void){
    regex_t regRep;
    char *regEx= "ne"; /* <-- Regular expression */
    char *txtPtr;
    size_t lineSize= maxSize-1;
    int er;
    txtPtr= (char *)malloc(maxSize);
    er= regcomp(&regRep, regEx, REG_EXTENDED | REG_NEWLINE);
    if (er != 0){
       printf("Error in regular expression\n");
       return 0; }
    while (getline(&txtPtr, &lineSize, stdin) > 0){
       er= regexec(&regRep, txtPtr, 0, NULL, 0);
       if (er == 0){
          printf("%s", txtPtr); /* <-- Action */
            }
       }
    regfree(&regRep);
    return 0;
 }
```

AWK

Dla pliku o następującej zawartości:

```
Jerzy Nawrocki   43089 I1
Jane  Kowalski   43780 I2
Adam  Malinowski 43990 I1
```

Z1. Uruchom i przeanalizuj poniższy program.

```
$4=="I1" { print $2, $1; }
```

Z2. Uruchom i przeanalizuj poniższy program.

```
$4=="I1"
```

Z3. Uruchom i przeanalizuj poniższy program.

```
  { print $2, $1; }
```

Z4. Uruchom i przeanalizuj poniższy program.

```
BEGIN { print "-----"; }
$4=="I2" { print $2, $1; }
END { print "*****"; }
```

Z5. Uruchom i przeanalizuj poniższy program.

```
END { print "*****"; }
$4=="I2" { print $2, $1; }
BEGIN { print "-----"; }
```

Z6. Dla poniższych danych:

```
If you
have
a hammer, everything
looks like a nail
```

Uruchom i przeanalizuj poniższy program.

```
  {total= total + NF;}
END {print "Fields: ", total;
     print "Rows: ", NR;}
```

Z7. Dla poniższych danych:

```
Jurek 10 Adam 200
Params: 3 40 -5
```

Uruchom i przeanalizuj poniższy program.

```
{ for (i=1; i<=NF; i++)
  if ($i ~ /^[0-9]+$/) num++;}
END {print "Found "num" numbers.";}
```

Z8. Dla poniższych danych:

```
1
22
.
A
H2O
```

uruchom i przeanalizuj poniższy program.

```
/^[0-9]/  {print "d..";}
/^.$/      {print $1;}
```

Z9. Dla poniższych danych:

```
1
22
.
A
H2O
```

uruchom i przeanalizuj poniższy program:

```
/^[0-9]/  { print "d..";
            next; }
/^.$/     { print $1; }
```