Отчет по 4 лабораторной работе.

Команда: Холодов А.В. Иванов А.А. ПИН-21

Задание Л4.з1. Разработайте программу, выводящую на стандартный вывод группу, номер и состав команды при помощи функции $puts()$ библиотеки libc.

Примечание: при работе в ОС MS Windows возможны проблемы с кодировкой русского языка. Если они возникли — используйте транслит или любые доступные вам способы настройки.

```
void task4_1(){
    char info[] = "Group: PIN-21 \nTeam: Ivanov Artem, Kholodov Artem";
    puts(info);
}
```

Вывод:

```
Task 1
Group: PIN-21
Team: Ivanov Artem, Kholodov Artem
```

Задание Л4.з2. Разработайте программу на языке C/C++, создающую, инициализирующую одинаковыми значениями и выводящую на экран при помощи функции libc $printf()$ массивы из $N = 5$ чисел:

- Ms из 16-битных целых чисел (0xCODE);
- Ml из 32-битных целых чисел (0x DEAD BEEF);
- Mq из 64-битных целых чисел (0x BECA CA01 FFED C0CA);
- Mfs из 32-битных чисел с плавающей запятой (значение x по варианту);
- Mfl из 64-битных чисел с плавающей запятой (значение x по варианту).

Выведите каждый из целочисленных массивов как в знаковом десятичном (d), так и в шестнадцатеричном (X) виде, чтобы убедиться, что короткие значения не расширены до 32 бит, а длинные — не усечены.

$x - \pi$ с максимально возможной для типа точностью

Реализация:

```
void task4_2(){
    short Ms[5];
    int Ml[5];
    long long Mq[5];
    float Mfs[5];
    double Mfl[5];
    for (int i = 0; i < 5; i++){
        Ms[i] = static_cast<unsigned short>(0xC0DE);
        Ml[i] = static_cast<unsigned int>(0xDEADBEEF);
        Mq[i] = static_cast<unsigned long long>(0xBECACA01FFEDC0CA);
        Mfs[i] = M_PI;
        Mfl[i] = M_PI;
    }

    printf("16 bit: %hd, %hd, %#hX, %#hX, %#hX\n",Ms[0],Ms[1],Ms[2],Ms[3],Ms[4]);
    printf("32 bit: %d, %d, %#X, %#X, %#X\n",Ml[0],Ml[1],Ml[2],Ml[3],Ml[4]);
    printf("64 bit: %lld, %lld, %#llX, %#llX, %#llX\n",Mq[0],Mq[1],Mq[2],Mq[3],Mq[4]);
    printf("32 bit float: %.16f, %.16f, %.16f, %.16f, %.16f\n",Mfs[0],Mfs[1],Mfs[2],Mfs[3],Mfs[4]);
    printf("64 bit double: %.16f, %.16f, %.16f, %.16f, %.16f\n",Mfl[0],Mfl[1],Mfl[2],Mfl[3],Mfl[4]);
}
```

Вывод:

```
Task 2
16 bit: -16162, -16162, 0XC0DE, 0XC0DE, 0XC0DE
32 bit: -559038737, -559038737, 0XDEADBEEF, 0XDEADBEEF, 0XDEADBEEF
64 bit: -4698721151270141750, -4698721151270141750, 0XBECACA01FFEDC0CA, 0XBECACA01FFEDC0CA, 0XBECACA01FFEDC0CA
32 bit float: 3.1415927410125732, 3.1415927410125732, 3.1415927410125732, 3.1415927410125732, 3.1415927410125732
64 bit double: 3.1415926535897931, 3.1415926535897931, 3.1415926535897931, 3.1415926535897931, 3.1415926535897931
```

Задание Л4.з3. Для каждого массива M введите с клавиатуры новое значение элемента $M[i]$, $i = 2$ при помощи функции libc $scanf()$. Проанализировав возвращённое $scanf()$ значение, определите корректность ввода; при необходимости отобразите сообщение об ошибке при помощи функции libc $puts()$.

Выведите массивы на экран снова, убедитесь, что элемент $M[i]$ приобрёл ожидаемое значение, а другие элементы массива не изменились.

Реализация:

```
35  void task4_3(){
36      short Ms[5];
37      int Ml[5];
38      long long Mq[5];
39      float Mfs[5];
40      double Mfl[5];
41      for (int i = 0; i < 5; i++){
42          Ms[i] = static_cast<unsigned short>(0xC0DE);
43          Ml[i] = static_cast<unsigned int>(0xDEADBEEF);
44          Mq[i] = static_cast<unsigned long long>(0xBECAC
45          Mfs[i] = M_PI;
46          Mfl[i] = M_PI;
47      }
48
49      //Ms
50      printf("New Ms[2]: ");
51      if (scanf("%hd", &Ms[2]) != 1) {
52          puts("Error");
53          while(getchar()!='\n');
54      }
55      else {
56          printf("New 16 bit: \n");
57          for (int i = 0; i < 5; i++){
58             printf("%d\t", Ms[i]);
59          }
60          printf("\n\n");
61      }
62
63      //Ml
64      printf("New Ml[2]: ");
65      if (scanf("%d", &Ml[2]) != 1){
66         puts("Error");
67         while(getchar()!='\n');
68      }
69      else {
70          printf("New 32 bit: \n");
71          for (int i = 0; i < 5; i++){
72             printf("%d\t", Ml[i]);
73          }
74          printf("\n\n");
75      }
76
```

```
77      //Mq
78      printf("New Mq[2]: ");
79      if (scanf("%lld", &Mq[2]) != 1){
80         puts("Error");
81         while(getchar()!='\n');
82      }
83      else {
84          printf("\nNew 64 bit: \n");
85          for (int i = 0; i < 5; i++){
86             printf("%lld\t", Mq[i]);
87          }
88          printf("\n\n");
89      }
90
91
92      //Mfs
93      printf("New Mfs[2]: ");
94      if (scanf("%f", &Mfs[2]) != 1){
95          puts("Error");
96          while(getchar()!='\n');
97      }
98      else {
99          printf("\nNew 32 bit float: \n");
100         for (int i = 0; i < 5; i++){
101            printf("%.16f\t", Mfs[i]);
102         }
103         printf("\n\n");
104     }
105
106     //Mfl
107     printf("New Mfl[2]: ");
108     if (scanf("%f", &Mfl[2]) != 1){
109         puts("Error");
110         while(getchar()!='\n');
111     }
112    else {
113         printf("\nNew 64 bit double: \n");
114         for (int i = 0; i < 5; i++)
115         {
116            printf("%.16f\t", Mfl[i]);
117         }
118         printf("\n");
119     }
120 }
121
```

Вывод:

```
Task 3
New Ms[2]: 13
New 16 bit:
-16162  -16162  13      -16162  -16162

New Ml[2]: 123
New 32 bit:
-559038737      -559038737      123     -559038737      -559038737

New Mq[2]: 45

New 64 bit:
-4698721151270141750    -4698721151270141750    45      -4698721151270141750    -4698721151270141750

New Mfs[2]: 65

New 32 bit float:
3.1415927410125732      3.1415927410125732      65.000000000000000      3.1415927410125732      3.1415927410125732

New Mfl[2]: k
Error
```

Задание Л4.з4. Для одного из массивов (по варианту) M введите с клавиатуры новое значение всех пяти элементов при помощи одного вызова функции libc $scanf()$. Проанализировав возвращённое $scanf()$ значение, определите корректность ввода; при необходимости отобразите сообщение о количестве не заданных элементов.

Выведите массив на экран снова.

Для Mfl;

Реализация:

```
void task4_4(){
    //Mfl
    double Mfl[5] = {M_PI, M_PI, M_PI, M_PI, M_PI};
    printf("New Mfl:\n");
    if (scanf("%lf %lf %lf %lf %lf", &Mfl[0], &Mfl[1], &Mfl[2], &Mfl[3], &Mfl[4]) != 5) {
        puts("Error.");
        while(getchar()!='\n');
    }
    puts("NewMFl: ");
    for (int i = 0; i < 5; i++)
    {
        printf("%.16f\t", Mfl[i]);
    }
    puts("\n");
}
```

Вывод:

```
Task 4
New Mfl:
12 3 56 76 4
NewMFl:
12.000000000000000      3.000000000000000      56.000000000000000      76.000000000000000      4.000000000000000
```

Задание Л4.з5. Введите с клавиатуры при помощи функций libc:

- слово (строку без пробелов) $s1$;
- строку, возможно, содержащую пробелы $s2$;
- слово $s3$ таким образом, чтобы принимающий его буфер гарантированно не переполнился (если буфер длины k — вводить не более $k-1$ символов, при необходимости добавляя завершающий нулевой символ).

Выведите на экран при помощи функций libc строки «*** $s1$ ***», «*** $s2$ ***», «*** $s3$ ***» (между звёздочками должна быть введённые строки, а не литералы s1-s3).

Реализация:

```
void task4_5(){
    char s1[100],s2[100],s3[5];

    scanf("%s\n",s1);
    scanf("%[^\n]s\n",s2);
    scanf("%4s\0\n",s3);

    printf("***%s***\n",s1);
    printf("***%s***\n",s2);
    printf("***%s***\n",s3);
}
```

Вывод:

```
Task 5
яблоко
гранатовый сок
код
***яблоко***
***гранатовый сок***
***код***
```

Задание Л4.з6. Выведите на экран при помощи функций libc массивы $Ms \ldots Mfl$ как таблицу из пяти строк и N столбцов (младшая цифра под младшей цифрой).

Реализация:

```
void task4_6(){
    short Ms[5];
    int Ml[5];
    long long Mq[5];
    float Mfs[5];
    double Mfl[5];
    for (int i = 0; i < 5; i++){
        Ms[i] = static_cast<unsigned short>(0xC0DE);
        Ml[i] = static_cast<unsigned int>(0xDEADBEEF);
        Mq[i] = static_cast<unsigned long long>(0xBECACA
        Mfs[i] = M_PI;
        Mfl[i] = M_PI;
    }

    printf("\n");
    for (int i = 0; i < 5; i++){
       printf("%23d", Ms[i]);
    }
    printf("\n");
    for (int i = 0; i < 5; i++){
       printf("%23d", Ml[i]);
    }
    printf("\n");
    for (int i = 0; i < 5; i++){
       printf("%23lld", Mq[i]);
    }
    printf("\n");
    for (int i = 0; i < 5; i++){
       printf("%23.16f", Mfs[i]);
    }
    printf("\n");
    for (int i = 0; i < 5; i++){
       printf("%23.16f", Mfl[i]);
    }
    printf("\n");
}
```

Вывод:

```
Task 6

                  -16162                  -16162                  -16162                  -16162                  -16162
                -559038737              -559038737              -559038737              -559038737              -559038737
  -4698721151270141750    -4698721151270141750    -4698721151270141750    -4698721151270141750    -4698721151270141750
      3.1415927410125732      3.1415927410125732      3.1415927410125732      3.1415927410125732      3.1415927410125732
      3.1415926535897931      3.1415926535897931      3.1415926535897931      3.1415926535897931      3.1415926535897931
```

Л4.3. Вопросы

1. Какие функции libc используются для форматированного ввода/вывода?
2. Как задаётся формат ввода/вывода для $scanf()/printf()$?
3. Как задаётся размер вводимых/выводимых данных для $scanf()/printf()$?

1.

Ввод и вывод информации осуществляется через функции стандартной библиотеки. Прототипы рассматриваемых функций находятся в файле `stdio.h`. Эта библиотека содержит функции

- `printf()` — для вывода информации
- `scanf()` — для ввода информации.

2.

- `'\n'` — перевод строки;
- `'\t'` — горизонтальная табуляция;
- `'\v'` — вертикальная табуляция;
- `'\b'` — возврат на символ;
- `'\r'` — возврат на начало строки;
- `'\a'` — звуковой сигнал.

3.

- `%d` — целое число типа `int` со знаком в десятичной системе счисления;
- `%u` — целое число типа `unsigned int`;
- `%x` — целое число типа `int` со знаком в шестнадцатеричной системе счисления;
- `%o` — целое число типа `int` со знаком в восьмеричной системе счисления;
- `%hd` — целое число типа `short` со знаком в десятичной системе счисления;
- `%hu` — целое число типа `unsigned short`;
- `%hx` — целое число типа `short` со знаком в шестнадцатеричной системе счисления;
- `%ld` — целое число типа `long int` со знаком в десятичной системе счисления;
- `%lu` — целое число типа `unsigned long int`;
- `%lx` — целое число типа `long int` со знаком в шестнадцатеричной системе счисления;
- `%f` — вещественный формат (числа с плавающей точкой типа `float`);
- `%lf` — вещественный формат двойной точности (числа с плавающей точкой типа `double`);
- `%e` — вещественный формат в экспоненциальной форме (числа с плавающей точкой типа `float` в экспоненциальной форме);
- `%c` — символьный формат;
- `%s` — строковый формат.