

Лабораторная работа №2

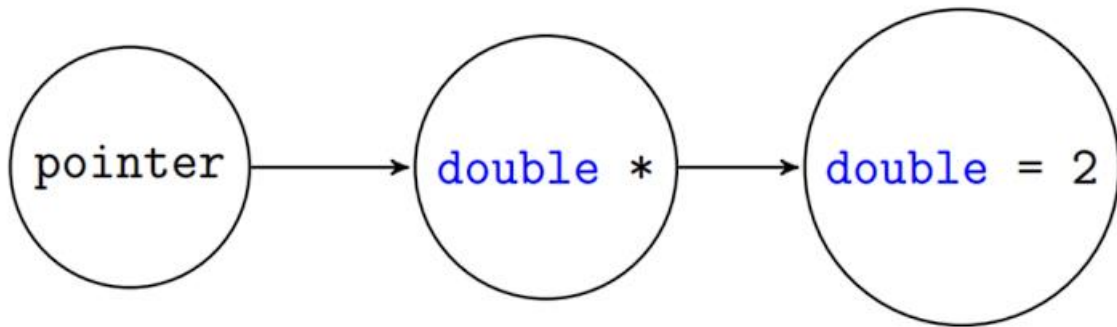
Тема Лабораторной работы: Указатели, арифметика указателей.

Задание 1.1

Постановка задачи:

Внутри функции `int main(void) { /*... */ }` определите указатель `double **pointer = NULL;`. Инициализируйте этот указатель адресом другого указателя типа `double *`, который указывает, в свою очередь, на переменную `double`. Используйте `pointer` для записи и чтения в эту переменную значения 2.

Математическая модель:



Список идентификаторов:

Имя	Тип	Смысл
ptr	double	Указатель на double
pointer	double	Указатель на указатель на тип double

Код программы:

```

1.c > main(void)
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(void) {
5      double *ptr = (double*)malloc(sizeof(double));
6      *ptr = 2;
7      double **pointer = &ptr;
8      printf("%lf\n", **pointer);
9      free(ptr);
10     return 0;
11 }

```

Результат выполненной работы:

```

PS C:\Users\gnev\OneDrive\
2.000000

```

Задание 1.2

Постановка задачи:

Напишите программу, которая складывает два числа с использованием указателей на эти числа.

Математическая модель:

$$A+B=C$$

Список идентификаторов:

Имя	Тип	Смысл
a	integer	Входящая переменная
b	integer	Входящая переменная
sum	integer	Результирующая переменная
ptr1	integer	Указатель на a
ptr2	integer	Указатель на b

Код программы:

```

C 2.c > main()
1  int main() {
2      int a, b, sum;
3      int *ptr1, *ptr2;
4
5      printf("Enter two numbers: ");
6      scanf("%d %d", &a, &b);
7
8      ptr1 = &a;
9      ptr2 = &b;
10
11     sum = *ptr1 + *ptr2;
12
13     printf("Sum is %d\n", sum);
14
15     return 0;
16 }

```

Результат выполненной работы:

```

Enter two numbers: 5 10
Sum is 15

```

Задание 1.3

Постановка задачи:

Напишите программу, которая находит максимальное число из двух чисел, используя указатели на эти числа.

Математическая модель:

$A > B$

Список идентификаторов:

Имя	Тип	Смысл
a	integer	Входящая переменная
b	integer	Входящая переменная
ptr1	integer	Указатель на a
ptr2	integer	Указатель на b

Код программы:

```

C 2.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int a, b;
6      int *ptr1, *ptr2;
7
8      printf("Enter two numbers: ");
9      scanf("%d %d", &a, &b);
10
11     ptr1 = &a;
12     ptr2 = &b;
13
14     if (*ptr1 > *ptr2) {
15         printf("%d > %d \n", *ptr1, *ptr2);
16     } else if (*ptr2 > *ptr1) {
17         printf("%d > %d \n", *ptr2, *ptr1);
18     } else {
19         printf("%d = %d \n", *ptr1, *ptr2);
20     }
21
22     return 0;
23 }

```

Результат выполненной работы:

```

Enter two numbers: 50 121
121 > 50

```

Задание 1.4

Постановка задачи:

Напишите программу, которая создаёт одномерный динамический массив из чисел с плавающей точкой двойной точности, заполняет его значениями с клавиатуры и распечатывает все элементы этого массива, используя арифметику указателей (оператор +), а не обычный оператор доступа к элементу массива - [].

Математическая модель:

$$A = [a_1, a_2, a_3, a_4, \dots, a_n]$$

Список идентификаторов:

Имя	Тип	Смысл
n	integer	Входящая переменная
i	integer	Параметр цикла
arr	double	Указатель на double
Ptr	double	Указатель на double

Код программы:

```

4.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int n, i;
6      double *arr, *ptr;
7
8      printf("Enter size : ");
9      scanf("%d", &n);
10
11     arr = (double*)malloc(n * sizeof(double));
12
13     printf("Enter elements :\n");
14     for (i = 0; i < n; i++) {
15         scanf("%lf", arr + i);
16     }
17
18     printf("elements of the array:\n");
19     for (ptr = arr; ptr < arr + n; ptr++) {
20         printf("%lf ", *ptr);
21     }
22
23     free(arr);
24
25     return 0;
26 }

```

Результат выполненной работы:

```

Enter size : 5
Enter elements :
1
2
3
4
5
elements of the array:
1.000000 2.000000 3.000000 4.000000 5.000000

```

Задание 1.5

Постановка задачи:

Вывести элементы динамического массива целых чисел в обратном порядке, используя указатель и операцию декремента ($--$).

Математическая модель:

$A = [1\ 2\ 3] \Rightarrow B = [3\ 2\ 1]$

Список идентификаторов:

Имя	Тип	Смысл
n	integer	Входящая переменная
i	integer	Параметр цикла
arr	integer	Указатель на int
ptr	integer	Указатель на int

Код программы:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int n, i;
6      int *arr, *ptr;
7
8      printf("Enter size: ");
9      scanf("%d", &n);
10
11     arr = (int*)malloc(n * sizeof(int));
12
13     printf("Enter elements:\n");
14     for (i = 0; i < n; i++) {
15         scanf("%d", arr + i);
16     }
17
18     printf("The reverse:\n");
19     for (ptr = arr + n - 1; ptr >= arr; ptr--) {
20         printf("%d ", *ptr);
21     }
22
23     free(arr);
24
25     return 0;
26 }
```

Результат выполненной работы:

```
Enter size: 5
Enter elements:
43 124 51 12 62
The reverse:
62 12 51 124 43
```

Задание 1.6

Постановка задачи:

Определите переменную целого типа `int a = 1234567890`; и выведите побайтово её содержимое на экран, используя указатель `char *`

Математическая модель:

$$a = b_1 + 256 * b_2 + 65536 * b_3 + 16777216 * b_4$$

Список идентификаторов:

Имя	Тип	Смысл
a	integer	Входящая переменная
ptr	char	Указатель на тип char переменной a

Код программы:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int a = 1234567890;
6      char *ptr = (char*)&a;
7
8      printf("Bytes:\n");
9      for (int i = 0; i < sizeof(int); i++) {
10         printf("%d ", *(ptr + i));
11     }
12
13     return 0;
14 }
```

Результат выполненной работы:

```
Bytes:
-46 2 -106 73
86 6 11 1
```

Задание 1.7

Постановка задачи:

Выделите память под двумерный динамический массив, используя массив указателей на строки (см. лекции). Затем освободите корректно оперативную память.

Математическая модель:

$$A(i,j) = x$$

Список идентификаторов:

Имя	Тип	Смысл
n_rows	integer	Входящая переменная
n_cols	integer	Входящая переменная
i	integer	Параметр цикла
j	integer	Параметр цикла
arr	integer	Указатель на тип int

Код программы:

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int n_rows, n_cols, i, j;
6      int **arr;
7      printf("Number of rows: ");
8      scanf("%d", &n_rows);
9      printf("Number of columns: ");
10     scanf("%d", &n_cols);
11
12     arr = (int**)malloc(n_rows * sizeof(int*));
13
14     for (i = 0; i < n_rows; i++) {
15         arr[i] = (int*)malloc(n_cols * sizeof(int));
16     }
17
18     for (i = 0; i < n_rows; i++) {
19         for (j = 0; j < n_cols; j++) {
20             arr[i][j] = i * j;
21         }
22     }
23
24     printf("Array:\n");
25     for (i = 0; i < n_rows; i++) {
26         for (j = 0; j < n_cols; j++) {
27             printf("%d ", arr[i][j]);
28         }
29         printf("\n");
30     }
31
32     for (i = 0; i < n_rows; i++) {
33         free(arr[i]);
34     }
35
36     free(arr);
37
38     return 0;
39 }

```

Результат выполненной работы:

```

Number of rows: 4
Number of columns: 5
Array:
0 0 0 0 0
0 1 2 3 4
0 2 4 6 8
0 3 6 9 12

```