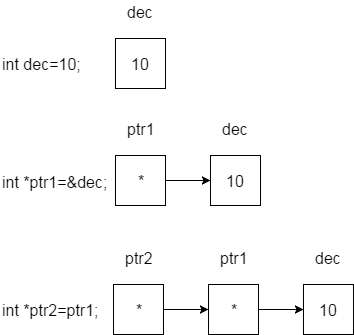
# Самостоятельная работа №7

# “Указатели”

## Задание 1

Каждой строке кода сопоставьте графическое представление.



int \*\*ref=&dec; ожидается указатель на указатель, а не на значение.

## Задание 2

Дано следующее описание переменных:

int \*p,\*q;

char \*r;

Какие из следующих операторов присваивания неправильны и

почему?

(а) p=q; (в) p=NULL; (д) q=\*p;

(б) q=r; (г) r=NULL; (е) \*p=NULL.

Неправильные операторы присваивания:

Б) Разный тип данных;

Д) Указателю присваивается не адрес переменной;

Е) Значение NULL может быть присвоено адресу, а не значению.

## Задание 3

Дано следующее описание переменных:

int \*p,\*q;

char \*r;

Какие из следующих операторов неправильны и почему?

(а) \*r=\*p; (б) \*q=(int)\*r;

(в) if (r!=NULL) (г) if (q>NULL)

\*r=\*NULL; \*q=\*p;

(д) if (q==p) (e) if (q!=r)

printf("%d",q); scanf("%c",r);

Неправильные операторы:

А) Адресу одного объекта нельзя присвоить адрес другого;

Г) С NULL можно сравнивать, используя только == или !=;

Е) Сравниваются данные разных типов.

## Задание 4

Выполните каждое из следующих заданий, используя для этой

цели только один оператор. Считайте объявленными переменные

value1 и value2 типа long, причѐм переменной value11 присвоено

значение 20000.

(а) Объявите указатель lPtr на объект данных типа long.

(б) Присвойте значение адреса переменной value1 указателю lPtr.

(в) Выведите значение объекта, на который ссылается lPtr.

(г) Присвойте значение объекта, на который ссылается lPtr, переменной

value2.

(д) Выведите значение value2.

(е) Выведите адрес value1.

(ж) Выведите значение адреса, находящееся в lPtr.

## Математическая модель

lPtr = &value1;

value2 = \*lPtr;

## Список идентификаторов

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип переменной | Смысл |
| lPtr | long | Указатель |
| value1 | long | Объект |
| value2 | long | Объект |

## Код программы

#include <stdio.h>

#include <locale.h>

void main() {

char \*locale = setlocale(LC\_ALL, "");

long value1=20000, value2;

long \*lPtr; //(а)

lPtr = &value1; //(б)

printf("Значение объекта, на который ссылается lPtr = %d\n", \*lPtr); //(в)

value2 = \*lPtr; //(г)

printf("Значение value2 = %d\n", value2); //(д)

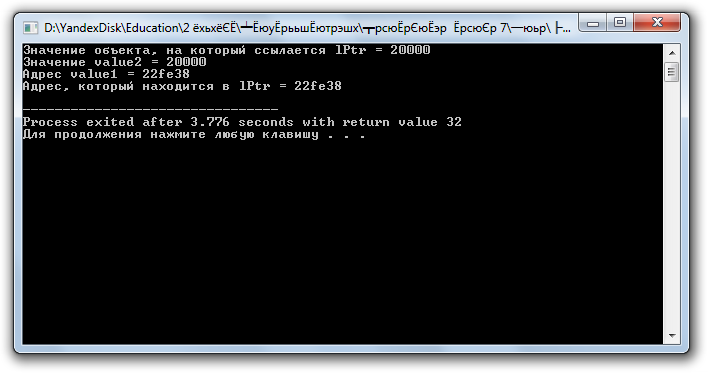
printf("Адрес value1 = %x\n", &value1); //(е)

printf("Адрес, который находится в lPtr = %x\n", lPtr); //(ж)

getch();

}

## Результат проделанной работы



## Задача 5

Найдите ошибку в каждом из следующих программных

фрагментов. Если ошибку можно исправить, пропишите, как это

сделать.

(а) int \*number;

printf("%d\n",\*number);

Нет адреса, на который ссылается указатель. Необходимо его прописать.

(б) float \*realPtr;

long \*integerPtr;

integerPtr=realPtr;

Разные типы данных. Необходимо все “свести” к одному типу.

(в) int \*x,y;

x=y;

Указателю нельзя присваивать число. Но можно присваивать адрес. Необходимо добавить & перед y.

(г) short \*numPtr,result;

void \*genericPtr=numPtr;

result=\*genericPtr+7;

Тип указателя void, поэтому ему нельзя присвоить адрес. Можно поменять тип указателя.

(д) float x=19.34;

float xPtr=&x;

printf("%f\n",xPtr);

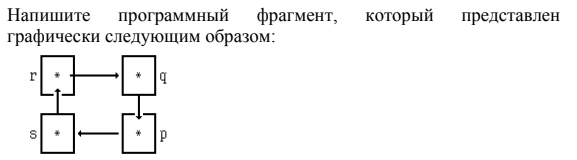
Адрес присваивается не указателю. Необходимо добавить \* перед xPtr.

(е) char \*s;

printf("%s\n",s);

Неверный формат вывода. Необходимо заменить %s для корректного вывода.

## Задача 6



## Программный фрагмент

int \*\*p,\*\*s,\*\*q,\*\*r;

p = &s;

s = &r;

r = &q;

q = &p;