```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int SumDigits(int number) {
  int sum = 0;

  while (number > 0) {

    sum += number % 10;

    number /= 10;

  }

  return sum;
}

int main(void) {
  setlocale(LC_ALL, "RU");

  int value = 0;

  scanf("%d", &value);

  printf("Сумма цифр числа %d: %d\n", value, SumDigits(value));

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

void IncrementAndPrint(void) {
  static int staticValue = 1;

  ++staticValue;

  printf("%d\n", staticValue);
}

int main(void) {
  setlocale(LC_ALL, "RU");

  IncrementAndPrint();
  IncrementAndPrint();
  IncrementAndPrint();

  return EXIT_SUCCESS;
}
```

```
int i;              // Объявление 1

void f(int i) {     // Объявление 2
  i = 1;
}

void g(void) {
  int i = 2;        // Объявление 3

  if (i > 0) {
    int i;          // Объявление 4

    i = 3;
  }

  i = 4;
}

void h(void) {
  i = 5;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

void UseLocal(void);         // Объявление функции
void UseStaticLocal(void);   // Объявление функции
void UseGlobal(void);        // Объявление функции

extern int x = 1;  // Глобальная переменная

int main(void) {
  setlocale(LC_ALL, "RU");

  int x = 5;  // Локальная переменная функции main
  printf("Значение локальной переменной x "
         "из внешней области main: %d\n", x);

  { // Начало нового блока
    int x = 7;  // Локальная переменная внутреннего блока
    printf("Значение локальной переменной x "
           "из внутренней области main: %d\n", x);
  } // Конец нового блока

  printf("Значение локальной переменной x "
         "из внешней области main: %d\n", x);

  UseLocal();        // Автоматический класс хранения локальной переменной x
  UseStaticLocal(); // Статический класс хранения локальной переменной x
  UseGlobal();       // Использование глобальной переменной
  UseLocal();        // Повторная инициализация локальной переменной x
  UseStaticLocal(); // Статическая локальная переменная x сохраняет предыдущее значение
  UseGlobal();       // Глобальная переменная также сохраняет значение

  printf("\nЗначение локальной переменной x в main: %d\n", x);

  return EXIT_SUCCESS;
}
```

```
// Функция UseLocal инициализирует локальную переменную x при каждом вызове
void UseLocal(void) {
  int x = 25; // Инициализируется каждый раз при вызове UseLocal

  printf("\nВ начале выполнения UseLocal "
         "локальная переменная x = %d\n", x);
  ++x;
  printf("В конце выполнения UseLocal "
         "локальная переменная x = %d\n", x);
}

// Функция UseStaticLocal инициализирует статическую локальную переменную x
// только при первом вызове. Значение x сохраняется между вызовами этой функции
void UseStaticLocal(void) {
  static int x = 50;  // Инициализируется единожды

  printf("\nВ начале выполнения useStaticLocal "
         "локальная статическая переменная x = %d\n", x);
  ++x;
  printf("В конце выполнения useStaticLocal "
         "локальная статическая переменная x = %d\n", x);
}

// Функция UseGlobal изменяет значение переменной x при каждом вызове
void UseGlobal(void) {

  printf("\nВ начале выполнения UseGlobal "
         "глобальная переменная x = %d\n", x);
  x *= 10;
  printf("В конце выполнения UseGlobal "
         "глобальная переменная x = %d\n", x);
}
```

Типовая структура простой программы

Директивы **`#include`**

Директивы **`#define`**

Определения типов

Объявления глобальных переменных

Объявления пользовательских функций

Определение функции **`main`**

Определения пользовательских функций

```
int *p;
```

```
int i, j, a[10], b[20], *p, *q;
```

```
int *p;
double *q;
char *r;
```

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main(void) {
  setlocale(LC_ALL, "RU");

  int *intPointer;
  double *doublePointer;
  char *charPointer;

  int x = 5;
  double y = 5.5;
  char z = 'a';

  intPointer = &x;

  printf("Значение intPointer: %p\n\n", intPointer);
  printf("Разыменованный указатель intPointer: %d\n\n", *intPointer);

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main(void) {
  setlocale(LC_ALL, "RU");

  int x = 5;
  double y = 5.5;
  char z = 'a';

  void *voidPointer = &x;

  printf("Значение voidPointer: %p\n\n", voidPointer);
  printf("Разыменованный указатель voidPointer: %d\n\n", *(int *)voidPointer);
  printf("sizeof(voidPointer): %d\n\n", sizeof(voidPointer));

  voidPointer = &y;

  printf("Значение voidPointer: %p\n\n", voidPointer);
  printf("Разыменованный указатель voidPointer: %g\n\n", *(double *)voidPointer);
  printf("sizeof(voidPointer): %d\n\n", sizeof(voidPointer));

  voidPointer = &z;

  printf("Значение voidPointer: %p\n\n", voidPointer);
  printf("Разыменованный указатель voidPointer: %c\n\n", *(char *)voidPointer);
  printf("sizeof(voidPointer): %d\n\n", sizeof(voidPointer));

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main(void) {
  setlocale(LC_ALL, "RU");

  int x = 5, *intPointer = &x;

  printf("Значение x: %d\n\n", x);
  printf("Разыменованный указатель intPointer: %d\n\n", *intPointer);

  *intPointer = -5;

  printf("Значение x: %d\n\n", x);
  printf("Разыменованный указатель intPointer: %d\n\n", *intPointer);

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main(void) {
  setlocale(LC_ALL, "RU");
  int x = 5;
  int *firstPointer, *secondPointer;
  firstPointer = &x;
  secondPointer = firstPointer;

  *firstPointer += 1;
  (*secondPointer)++;

  printf("Значение x: %d\n\n", x);

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main(void) {
  setlocale(LC_ALL, "RU");
  int x = 5, y;
  int *firstPointer, *secondPointer;
  firstPointer = &x;
  secondPointer = &y;

  *secondPointer = *firstPointer;

  printf("Значение y: %d\n\n", y);

  return EXIT_SUCCESS;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

void FindMaxAndMinInArray(int array[], int size, int *max, int *min);

int main(void) {
  setlocale(LC_ALL, "RU");

  int array[] = {5, 7, -2, 34, 0};

  int size = sizeof(array) / sizeof(array[0]);

  int max = 0, min = 0;

  FindMaxAndMinInArray(array, size, &max, &min);

  printf("Max: %d\nMin: %d\n", max, min);

  return EXIT_SUCCESS;
}

void FindMaxAndMinInArray(int array[], int size, int *max, int *min) {
  *max = *min = array[0];

  for (int i = 0; i < size; ++i) {
    if (array[i] > *max) {
      *max = array[i];
    }
    else if (array[i] < *min) {
      *min = array[i];
    }
  }
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

void Zeroing(int *ptr);

int main(void) {
  setlocale(LC_ALL, "RU");

  int value = 8;

  Zeroing(&value);

  printf("value: %d\n", value);

  return EXIT_SUCCESS;
}

void Zeroing(int *ptr) {
  printf("value: %d\n", *ptr);
  *ptr = 0;
}
```

```c
void Zeroing(const int *ptr);
void Zeroing(int* const ptr);
void Zeroing(const int* const ptr);
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int* MaxNumber(int *a, int *b);

int main(void) {
  setlocale(LC_ALL, "RU");

  int x = -325, y = 1;

  int* max = MaxNumber(&x, &y);

  printf("Наибольшее значение: %d\n", *max);

  --(*max);

  printf("y = %d\n", y);

  return EXIT_SUCCESS;
}

int* MaxNumber(int *a, int *b) {
  return *a > *b ? a : b;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int* GetMiddle(int array[], int size);

int main(void) {
  setlocale(LC_ALL, "RU");

  int array[] = {5, 7, -2, 34, 0};

  int size = sizeof(array) / sizeof(array[0]);

  int *ptr = GetMiddle(array, size);

  printf("Центральный элемент: %d\n", *ptr);

  *ptr = *ptr + *ptr;

  printf("array[%d] = %d\n", size / 2, array[size / 2]);

  return EXIT_SUCCESS;
}

int* GetMiddle(int array[], int size) {
  return &array[size / 2];
}
```