数据存储2.pdf
516.71KB

1.请简述大端字节序和小端字节序概念 并 设计一个小程序来判断当前机器的字节序

```c
int check_sys(){
int a = 1;
char* p = (char*)&a; //char* 步长为1字节 取出a的地址
return *p;
//返回1 小段
//返回0 大端
}

int main()
 {
 //返回1 小段
 //返回0 大端
 int ret = check_sys();
 if(ret == 1)
 {
 printf("小端");
 }else{
 printf("大端");
 }
 return 0;

 }
```

2.输出是什么?

```c
#include <stdio.h>
int main()
{
 char a = -1;
 //-1在内存中存储的是补码
 //10000000000000000000000000000001 原码
 //11111111111111111111111111111110 反码
 //11111111111111111111111111111111 补码
 //11111111 a中存储的
  //11111111 b中存储的
  //11111111 c中存储的
```

```
 signed char b = -1;
 unsigned char c = -1;
 printf("a=%d b=%d c=%d", a,b,c);
 //整形打印
 //a整形提升 11111111111111111111111111111111 空位补1 变成反码 算出原码为-1
 //b同a
 //c为unchar整型提升的时候高位补0 变成00000000000000000000000011111111 也就是255
 return 0;
}

//-1 -1 255
```

3.输出是什么

```
#include <stdio.h>
int main()
{
 char a = -128; //char 一个字节 八个bit位
 //原码 10000000 00000000 00000000 10000000
 //反码 111 11111 11111111 11111111 01111111
 //补码 11111111 11111111 11111111 10000000
 //存储到a中 10000000
 printf("%u",a);
 //%u打印十进制无符号数字 打印时原码反码补码相同
 //整型提升 11111111 11111111 11111111 10000000
 return 0;
}
//4294967168
```

4.按照补码的形式进行运算 最后格式化为有符号整数

```
int i = -20
//原码:  10000000 00000000 00000000 00010100
//反码 11111111 11111111 11111111 11101011
//补码 11111111 11111111 11111111 11101100
unsigned int j = 10;
//补码 00000000 00000000 00000000 00001010
printf("%d", i+j);
//补码相加 11111111 11111111 11111111 11110110
//-1取反得到原码-10
```

5.

```
int main()
{
```

```
 char a[1000];
 int i;
 for(i=0;i<1000;i++)
 {
 a[i] = -1 - i;
 }
 printf("%d", strlen(a));
  return 0;
}
//255
```

6.

```
#include <stdio.h>
int main()
{
 unsigned char i;
 for(i=0;i<=255;i++)
 {
 printf("hello world\n");
 }
 return 0;
}

//死循环 uchar i永远小于255
```

7.float 和 double类型