写出下列程序的运行结果

```
int main()
{  const char *c[]={"John learn C++ language",
                    "Be well!", "You", "Not very"};
   const char **p[]={c+3, c+2, c+1, c};
   const char ***pp=p;
   cout << (**++pp);
   cout << (*--*++pp+4);
   cout << (*pp[-2]+3);
   cout << (pp[-1][-1]+2);
   cout << endl;
   return 0;
}
```

注：直接在本文件上作答，画出程序执行过程的内存变化即可

★ 首先画出三句定义语句结束后内存中各变量的所占空间及初值

★ 每个执行语句的每一步执行完成后的内存中各变量的所占空间及值

★ 每步变化一个页面(例：**++pp，要三页)

★ 不允许手写在纸上，再拍照贴图

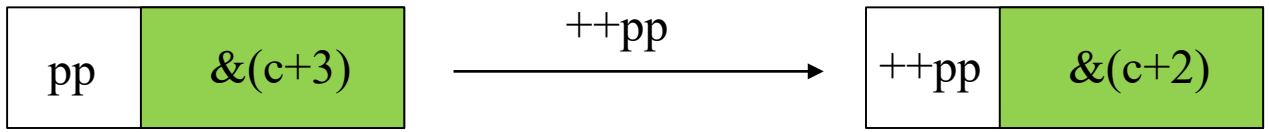
★ 允许在各种软件工具上手写完成，再截图贴图

★ 转换为pdf后提交

写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (**++pp);
```

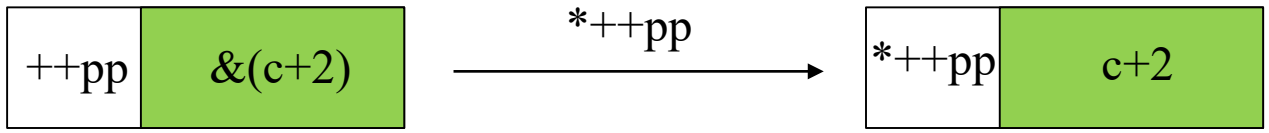
把p赋给pp，那么pp本身的值为p，而p是数组的地址，pp指向c+3的地址++pp是把指针向后移动一个，指向c+2的地址



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (**++pp);
```

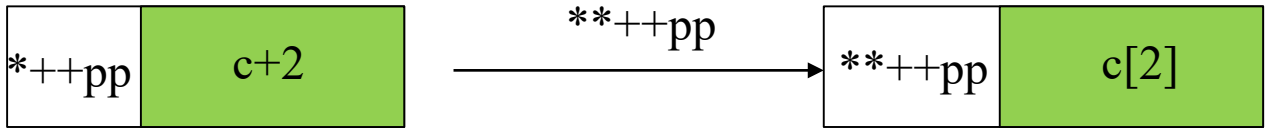
++pp现在指向c+2这个元素，*++pp则为c+2这个值



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (**++pp);
```

*++pp现在是c+2，**++pp则是c+2这个地址取内容，c+2是指针数组里指向You字符串的指针的地址，所以*取内容就是指向它的指针，而cout这个指针就是输出这个字符串You

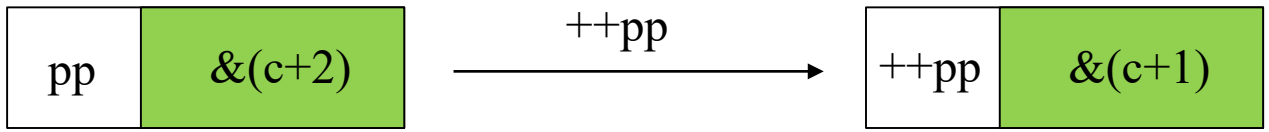


写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*--*++pp+4);
```

把p赋给pp，那么pp本身的值为p+1，而p是数组的地址，pp指向c+2的地址

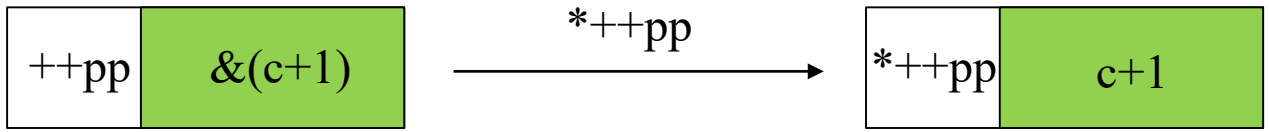
++pp是把指针向后移动一个，指向c+1的地址



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*--*++pp+4);
```

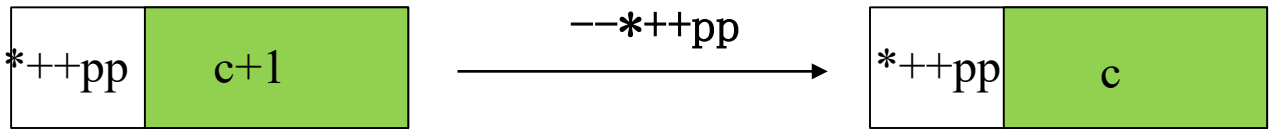
++pp现在指向c+1这个元素，*++pp则为c+1这个值



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*--*++pp+4);
```

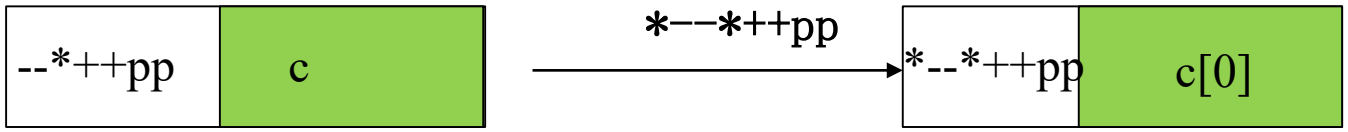
++pp现在是c+1，--++pp则为c这个值



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*--*++pp+4);
```

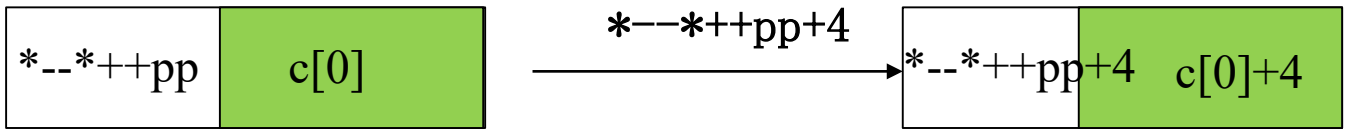
--*++pp现在是c，*--*++pp则为c本身这个地址取内容，则为指向John的指针



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*--*++pp+4);
```

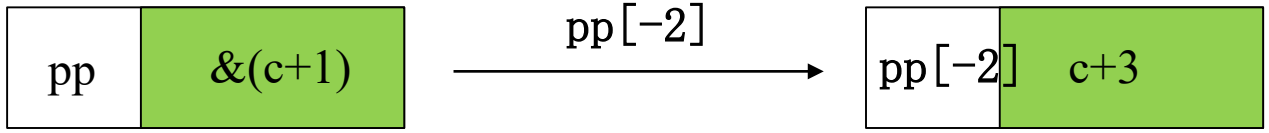
--++pp为指向John的指针，则*--*++pp+4为指针向该字符串后移4位，指向John和learn之间的空格,cout的话是输出 learn C++ language



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*pp[-2]+3);
```

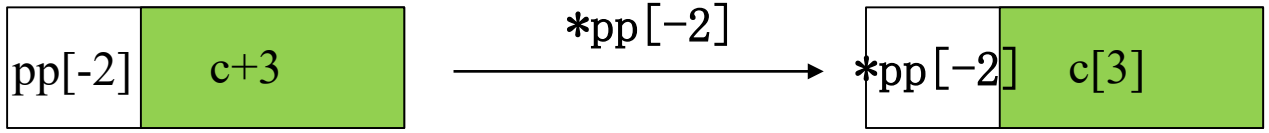
把p赋给pp，那么pp本身的值为p+2，而p是数组的地址，pp指向c+1的地址
pp[-2]是把指针不移动，只是*(pp-2)



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*pp[-2]+3);
```

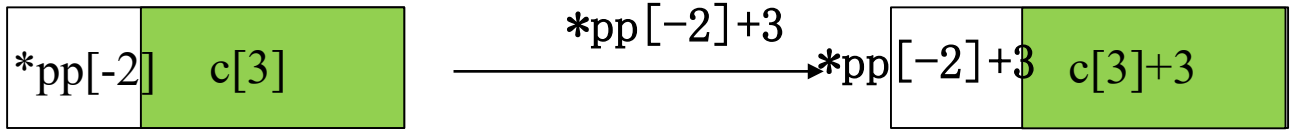
pp[-2]是c+3，取内容则为指向Not very的指针



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (*pp[-2]+3);
```

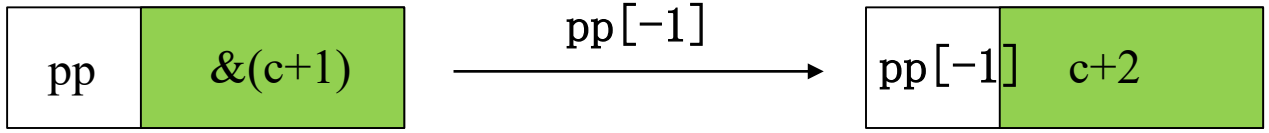
*pp[-2]是指向Not very的指针，则*pp[-2]+3为指向Not和very之间的空格，那么cout之后是输出的 very



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (pp[-1][-1]+2);
```

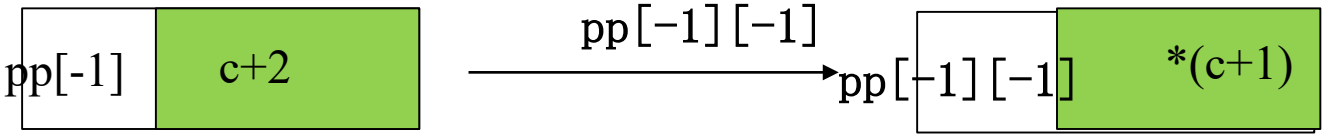
把p赋给pp，那么pp本身的值为p+2，而p是数组的地址，pp指向c+1的地址
pp[-1]是把指针不移动，只是*(pp-1)



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (pp[-1][-1]+2);
```

pp[-1]是c+2，而pp[-1][-1]是把c+2减1为c+1取内容，就是指向Be well的指针



写出下列程序的运行结果

```
const char *c[]={"John learn C++ language",
                                "Be well!", "You", "Not very"};
const char **p[]={c+3, c+2, c+1, c};
const char ***pp=p;
cout << (pp[-1][-1]+2);
```

pp[-1][-1]是指向Be well的指针，+2向后移动两个，指向Be和well之间的空格，那么cout之后就是输出 well