



## 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，分三步计算，需要三页)
  - ★ **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

```
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;
```

```
}
```

| *c   |      |
|------|------|
| 2000 | 3000 |
| 2004 | 3024 |
| 2008 | 3033 |
| 2012 | 3037 |

|      |         |
|------|---------|
| 3000 | John... |
| 3024 | Be....  |
| 3033 | You     |
| 3037 | Not.... |

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

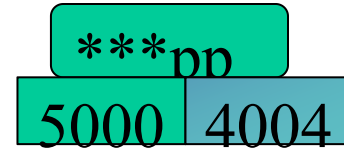
| ***pp |      |
|-------|------|
| 5000  | 4000 |



## 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;
}
```



++pp: pp指向了p[1]



# 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;
}
```

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4004 |

\*++pp:取到了地址2008



# 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;
}
```

\*c

2008 3033

3033 You

\*\*p

4004 2008

\*\*\*pp

5000 4004

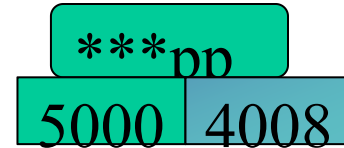
\*\*++pp: 取到了地址3033  
输出: You



## 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;
}
```



`++pp`: `pp` 指向了 `p[1]`



# 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;
}
```

| *c   |      |
|------|------|
| 2000 | 3000 |
| 2004 | 3024 |
| 2008 | 3033 |

|      |        |
|------|--------|
| 3024 | Be.... |
| 3033 | You    |

\*++pp 取值p[1]:2008

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |

| ***pp |      |
|-------|------|
| 5000  | 4004 |



# 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;
```

```
}
```

\*c

|      |      |
|------|------|
| 2004 | 3024 |
| 2008 | 3033 |

|      |        |
|------|--------|
| 3024 | Be.... |
| 3033 | You    |

--\*++pp 取2008地址减一:2004

\*\*p

|      |      |
|------|------|
| 4004 | 2008 |
|------|------|

\*\*\*pp

|      |      |
|------|------|
| 5000 | 4008 |
|------|------|





# 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;
}
```

\*c

|      |      |
|------|------|
| 2004 | 3024 |
| 2008 | 3033 |

|      |        |
|------|--------|
| 3024 | Be.... |
| 3033 | You    |

\*--\*++pp 解2004的地址:3024

\*\*p

|      |      |
|------|------|
| 4004 | 2008 |
|------|------|

\*\*\*pp

|      |      |
|------|------|
| 5000 | 4008 |
|------|------|



## 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;
}
```

|      |        |
|------|--------|
| 3024 | Be.... |
|------|--------|

\*--\*++pp+4 从第四个开始输出:  
ell!



# 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;
}
```

pp[-2]:取到了p向后两个基类型的地址 3992

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4000 |



# 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;
}
```

\*pp[-2]:取到了p向后两个基类型的地址的值 -- “未知”

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4000 |



# 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;
}
```

\*pp[-2]+3:未定义行为输出, 返回-192341523

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4000 |



# 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;
}
```

pp[-1]:取到了p向后1个基类型的地址的内容

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4000 |



# 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;
}
```

pp[-1][-1]:取到了p向后1个基类型的地址的内容的  
向下一个基类型的内容

| **p  |      |
|------|------|
| 4000 | 2012 |
| 4004 | 2008 |
| 4008 | 2004 |
| 4012 | 2000 |

| ***pp |      |
|-------|------|
| 5000  | 4000 |



## 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;
}
```

pp[-1][-1]+2:取到了p向后1个基类型的地址的内容的向后一个基类型的内容再向后两个基类型

未定义操作

返回-1073741819