提示：答案请填写在答题卡上，通过网络学堂作业提交电子版答题卡。

一、单选题（共 10 道题，每道题 3 分，总计 30 分）

(1)以下所有字符数组中，两次调用 read 函数时均输入 hello 并回车，一定能正常输出 hello 的数组有____个。

```
#include <stdio.h>
void read(char s[])
{    int i=0;
     char c;
     while((c=getchar()) != '\n')
          s[i++] = c;
}
void test()
{    char str1[10];
     read(str1);
     static char str2[10];
     read(str2);
     char str3[10] = {'h', 'e' ,'l', 'l', 'o'};
     char str4[] = "hello";
     char str5[] = {'h', 'e' ,'l', 'l', 'o'};
     printf("%s\n%s\n%s\n%s\n%s\n",str1,str2,str3,str4,str5);
}
int main()
{    test();
     return 0;
}
```

(A) 1　　(B) 2　　(C) 3　　(D) 4

全局或静态 char 数组（静态存储区）
未初始化的元素会被自动初始化为 0（即 '\0'）。

局部 char 数组（栈内存）
未初始化的元素的值是未定义的（垃圾值）

部分初始化的 char 数组
如果显式初始化部分元素，剩余元素会被自动填充为 0。

动态分配的 char 数组（堆内存）
使用 malloc/new 分配时，元素的值是未定义的。

使用 calloc 时，元素会被初始化为 0

所以第一题中2，3，4可以，选C

(2) 以下说法正确的是_____。

```
#include <stdio.h>
int main( )
{    struct
     { int a[2];
     }arr[  ] = {{1}, {2}};
     printf("%d %d %d %d", arr[0].a[0], arr[0].a[1], arr[1].a[0], arr[1].a[1]);
     return 0;
}
```

感觉这个卷子有点离谱啊，也许是我自学的原因，写的代码都比较规范，第二题创建了一个匿名结构体数组，数组里有两个结构体，每个还只初始化一个元素，另一个未初始化，对于另一个未初始化的值：如果结构体或数组只初始化部分成员，剩余部分会被自动初始化为 0（适用于全局变量或静态存储区）;但如果是局部变量，未初始化的部分可能是未定义的值（垃圾值），但现代编译器（如 gcc、clang）通常也会初始化为 0 以提高安全性。所以到底选B还是D呢，自学的人是搞不清楚出题人怎么想的，TMD大学出题还像高中这么偏吗？原则上讲B有道理，但实际上在多数编译器上结果是D,既然如此,考试时肯定选D,实际上谁会写这种神人代码，纯为出题而出题

(A) 编译错误，因为缺少结构体名　　(B) 编译错误，因为该结构数组的初始化有问题
(C) 没有编译错误，它会打印 1 2 0 0　　(D) 没有编译错误，它会打印 1 0 2 0

(3) 以下程序得到的结果为_____。

```
#include <stdio.h>
#define N 3
#define Y(n)    ((N+1)*n)
int main()
{ int z;
   z = N + Y(5+1);
   printf("%d",z);
}
```

(A) 24　　(B) 27　　(C) 48　　(D) 42

本题选A,原因是宏定义时是替换,3+(3+1)*5+1=24
嘿嘿,怎么样呢

(4) 有如下程序，该程序执行结果是_____。

```
#include <stdio.h>
int main( )
{ int x=4;
   do { printf("%d", x- -);
        }while(!x);
}
```

(A) 4321　　(B) 陷入死循环　　(C) 4　　(D) 不输出任何内容

没什么好说,选C

(5) 有如下程序，该程序的输出结果是_____。

```
#include <stdio.h>
int Cube(int i)
{      return (i * i * i) ;
}
int main( )
{    int i = 0;
     i = Cube(i);
     for(   ;   i<2;   i++)
     {    int j = 1;
          j += Cube(i);
```

奇怪的题，选B

C:指针变量的特性
char *s 定义的是一个指向字符的指针变量，它的值（即指向的地址）是可以被修改的。当执行 s = "china"; 时，实际上是将字符串常量 "china" 在内存中的首地址赋值给了指针变量 s，让 s 指向这个字符串的起始位置。字符串常量的本质:字符串常量 "china" 在内存中以字符数组的形式存储（末尾隐含 \0 结束符），其值是该数组的首地址。因此，s = "china" 本质上是地址赋值，即将字符串的首地址存入指针变量 s，这符合指针变量的操作规则。与数组名的区别数组名是常量指针（地址不可修改），而指针变量是变量指针（地址可修改）。例如：数组名 str 不能执行 str = "china"（数组名不可赋值）；
指针变量 s 可以执行 s = "china"（指针变量的值可修改）。

```
    }
    printf("%d", i);
    return 0;
}
```

A:数组在定义时必须指定大小（或通过初始化列表推断），否则无法分配内存。主调函数main中若使用数组名作为实参，该数组必然是已定义的（需明确大小），否则会报错。此说法正确。

(A) 0　　(B) 2　　(C) 3　　(D) 1

(6) 若使用一维数组名作函数实参，则以下正确的说法是_____。

(A) 必须在主调函数中说明此数组的大小　　(B) 实参数组名与形参数组名必须一致

(C) 实参数组类型与形参数组类型可以不匹配　　(D) 在被调用函数中，不需要考虑形参数组的类型

(7) 下面判断正确的是_____。

(A) char *a="china";等价于 char *a; *a="china";　　(B) char c[4]="abc",d[4]="abc";等价于 char c[4]=d[4]="abc";

(C) char *s="china";等价于 char *s; s="china";　　(D) char str[10]={"china"};等价于 char str[10];str[]={"china"};

(8) 有程序段如下，不能表示为地址的是_____。

选项 D：&(x+1);因为x+1是一个表达式，其结果是一个临时计算值（x的值为 3，x+1的结果为 4），该结果存储在 CPU 寄存器或临时内存中，没有固定的内存地址，属于右值 **（不能被取地址）。因此&(x+1)是非法的，无法表示地址。

```
int x=2, *p;
p=&x;
x=x+1;
```

(A) &x　　(B) p　　(C) &p　　(D) &(x+1)

(9) 以下能对某函数内的二维数组 a 中的所有元素进行正确初始化的语句是 B 。

(A) int a[2][4]={ {1, 2, 3}, {4, 5, 6} }　　(B) int a[ ][3]={ {1, 2, 3}, {4, 5, 6} }

(C) int a[2][4]={ {1, 2, 3, 4}, {5, 6} }　　(D) int a[ ][3]={ {1, 2, 3}, { },{4, 5} }

(10) 下列程序运行结果是_____。 B

s[2]--,先使用后--，先使用%c1=c；然后把c改成b

```
#include <stdio.h>
int main( )
{char s[10]={"abc"};
  printf("c1=%c, c2=%s\n", s[2]- -, s);
  return 0;
}
```

(A) c1=b,　c2=abb　　(B) c1=c,　c2=abb　　(C) c1=b,　c2=abc　　(D) c1=c,　c2=abc

二、多选题（共 5 题，每题每选错一个扣 1 分，封顶扣 2 分，全选对满分 10 分）

(1) 以下语句中肯定不能作为 switch 语句的 case 标签的是BC,需要是整形常量表达式。

(A) 'a'　　(B) 3.14　　(C) ival // ival 为一个已经定义好的变量　　(D) false

(2) 以下初始化正确的是 AB 。

(A) int array1[2][4] = {1, 2, 3, 4, 5, 6, 7, 8};　　(B) int array2[ ][4] = {1, 2, 3, 4, 5, 6, 7, 8};

(C) int array3[2][ ] = {1, 2, 3, 4, 5, 6, 7, 8};　　(D) int array4[ ][ ] = {1, 2, 3, 4, 5, 6, 7, 8};

(3) 以下关于 typedef 的叙述不正确的是_AD,typedef为已有类型定义别名，不能用于定义变量（定义变量需用类型名本身,如int a)。

(A)用 typedef 可以定义各种类型名，也可以用来定义变量　　(B)使用 typedef 便于程序的通用

(C)用 typedef 只是将已存在的类型用一个新名字来代表　　(D)用 typedef 可以增加新类型

(4) 设有以下定义：int a[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};　　int (*ptr)[4]=a,　*p=a[0];

则下列表达式中能正确表示数组元素 a[2][1]的表达式有_AB_。

(A) *(p+9)　　(B) *(*(a+2)+1)　　(C) (*ptr+2)+1　　(D) *((*ptr+2)[1])

(5) 已知：int a, *p=&a; 则为了得到变量 a 的值，下列正确的表达式有__BD。

(A) *p[0]　　(B) *p　　(C) &*a　　(D) *&a

三、填空题（共 14 道题，每道题 3 分，总计 42 分）

(1) 以下程序的输出结果为_4_。

```
#include <stdio.h>
int main( )
{    int a[5] = {0,1,2,3,4}, b, *p;
     p = a+1;
     b = *p;
     p += 1;
     b += 2;
     *p += 1;
     (*p) += 1;
     p = &a[1];
     a[p-a]+=b;
     printf("%d", a[2]);
     return 0;
}
```

(2) 以下程序的输出结果为__7__。

```
#include <stdio.h>
int func(int a)
{   int b = a++;
    return a;
}
int main( )
```

```
{    int k = 3;
        printf("%d", func(k) + k);
     return 0;
}
```

(3) 以下程序的输出结果为 _____b_____。

```
#include <stdio.h>
int main( )
{   int a = -30;
    char s[10] = "ab\0";
    printf("%s", s+1);
    return 0;
}
```

(4) 以下程序的输出结果在输入为 126 时为 _____15_____。

```
#include <stdio.h>
int n;
int main( )
{   int x=0;
 scanf("%d", &n);
    for(int i=2;  i*i<=n;  i++)
    { while(n%i == 0)
       { n=n/i;
     x=x+i;
       }
    }
    if(n>1)
        printf("%d", x+n);
    return 0;
}
```

(5) 以下程序的输出结果为_____24_____。

```
#include <stdio.h>
#include "stdlib.h"
int x=0;
void swap(char *a, char *b)
{   char *t;
    t = a;
    a = b;
    b = t;
}
void calc(char *s, int a, int b)
{
 if(a == b)
      x=x+atoi(s);
   else
   { for(int i=a; i<=b; i++)
     {   swap(&s[i], &s[a]);
         calc(s, a+1, b);
         swap(&s[a], &s[i]);
     }
   }
}
int main( )
{   char s[3] = "12";
    calc(s, 0, 1);
     printf("%d",x);
    return 0;
}
```

这两个swap纯迷惑人的，屁用没有，改变形参，对实参无影响

(6) 以下 C 语言程序的输出结果是 _____21_____。

```
#include <stdio.h>
struct s
{
   int x,y;
} data[2] = {10, 100, 20, 200};
int main( )
```

```
    {     struct s *p=data;
          p++;
          printf("%d", ++(p->x));
    }
```

(7) C 语言中，下面程序的输出结果是 5。

```
#include <stdio.h>
int fib(int   n)
{ if ((n= =0) || (n= =1))
      return 1;
   return (fib(n-1) + fib(n-2));
}
int main( )
{ printf("%d", fib(4));
   return 0;
}
```

(8) 以下 C 语言程序的输出结果是 2。

```
#include <stdio.h>
void swap(int *a, int b)
{ int m, *n;
   n=&m;
   *n=*a;
   *a=b;
   b=*n;
}
int main( )
{ int x=8, y=1;
   swap(&x, y);
   printf("%d", x+y);
   return 0;
}
```

(9) 以下 C 语言程序的输出结果是 20224396

```
#include <stdio.h>
void fun(char str[ ])
{int i, j=0;
 for (i=0;   str[i];   i++)
    if ( str[i]>='0'   &&   str[i]<='9')
          str[j++]= str[i];
 str[j]='\0';
}
int main( )
{ char str[100]="By the end of February of 2022, the total number of the teaching staff in THU has reached 4396.";
   fun(str);
   printf("%s", str);
   return 0;
}
```

(10) 以下 C 语言程序的输出结果是 7。

```
#include <stdio.h>
int gcd(int p,int q)
{   if(p= =q) return p;
     else if(p>q )
        return gcd(p - q, q);
     else
        return gcd(p, q - p) ;
}
int main( )
{ int m=2, n=3, min, max;
   max=gcd(m, n);
   min=m*n/max;
   printf("%d", max+min);
   return 0;
}
```

(11) 当从键盘输入 happy 后，以下代码的输出结果是 happy。

```
#include "stdio.h"
#include "string.h"
int main( )
{       char str[14]={"I am "};
```

```
        strcat(str, "sad!");
        scanf("%s", str);
        printf("%s", str);
}
```

使用 %s 读取字符串
scanf 会：读取 'h' 'a' 'p' 'p' 'y'（遇到 \n 停止）。自动在末尾添加 '\0'，形成合法的C字符串。\n 留在输入缓冲区（未被读取），所以看到的不是happysad!

(12) 以下程序的输出结果是_____5_____。

```
#include <stdio.h>
#define N 5
int main( )
{   int i, j, n=N, k,x=0;
    int num[N]={2, 3, 0, 2, 0};
    for(i=0;  i<n-1;  i++)
        for(j=i+1;  j<n;  j++)
            if(num[i] = = num[j])
             {for(k=j;  k<n-1;  k++)
                num[k]=num[k+1];
                - -n;
                - -j;
             }
    for(int i=0;  i<n-1;  i++)
        x=x+num[i];
    printf("%d", x+num[i]);
    return 0;
}
```

(13) 以下程序的输出结果是_____5_____。

```
#include <stdio.h>
#include <string.h>
int main( )
{   char str[ ]= "I love you China";
    int count=0;
    int len=strlen(str);
    for(int i=(len-1);  i>=0;  i- -)
    {   if(str[i] != ' ')
            count++;
        else
            break;
    }
    printf("%d",count);
    return 0;
}
```

(14) 以下程序的输出结果是___24___。

```
#include <stdio.h>
int f(int a)
{   int b=1;
    static int c=3;
    b++;
    c--;
    return (a+b+c);
}
int main()
{   int a=5,i,x=0;
    for(i=0;i<3;i++)
       x=x+f(a);
    printf("%d",x);
    return 0;
 }
```

四、 程序填空题

1. 从键盘上输入n（1<n<10），按样例格式打印n行由字符 * 构成的W形图案(本题目8分，每个空2分)。

【输入形式】从键盘输入整数n (1<n<10)，输入不合法时可重新输入。
【输出形式】输出用字符*构成的W形图案，样式按右下面的样例，第1行的起始字符位于第1列。如：输入为4时，输出如右图所示（图中首行为输入）。

```
#include <stdio.h>
int main( )
{   int n, i, j, k, a, b, c;
    do{   printf("请输入正整型数n,请注意1<n<10 \n");
          scanf("%d", &n);
      }              (1)_____ ;
  for(i = 0; i < n; i++)
    {   for(j= 0;  j<i;  j++)
              printf(" ");
                (2)        ;
          for(k = 0;  k < 2*(n-i)-3;  k++)
              printf(" ");
          if(i==0)
              printf("*");
            else if( i==(n-1) )
            {
            }
           else
             {  printf("*");
                for(a = 0;        (3)_____ ;  a++)
                    printf(" ");
                printf("*");
             }
        for(b = 0;  b< 2*(n-i)-3;  b++)
            printf(" ");
        if(i==(n-1))
            for(c = 0;  c< 2*n-3;  c++)
                printf(" ");
        ____(4)____ ;
    }
    return 0;
}
```

while(n<=1||n>=10);

printf("*");

a<i

printf("*\n");

2. 字符串的一个子序列是原始字符串删除一些（也可以不删除）字符而不改变剩余字符相对位置形成的新字符串。（例如，"ace"是"abcde"的一个子序列，而"aec"不是），本程序要求依据提示分别输入两个字符串S和T，判断第二个字符串T是否是满足上述定义的字符串S的子序列(本题目10分，每个空2分)。

```
#include <stdio.h>
#include <    (1)    >//添加字符串头文件
int Subsequence(char s[ ], char t[ ])
{   int m, n, i=0, j=0;
    n = strlen(s);
    m = strlen(t);
    if (m>n)
        return 0;
    while (          (2)_____ )
    {   if(t[i]==s[j])
                ____(3)_____ ;
        j=j+1;
    }
    if (        (4)_____)
        return 1;
    return 0;
}

int main( )
{   int Subsequence(char s[ ], char t[ ]);
    char s[30], t[30];
    printf("请输入待匹配的字符串S：");
```

string.h

j<n&&i<m

i=i+1

i==m

```
    scanf("%s", s);
    printf("请输入待匹配的字符串T：");
    scanf("%s", t);
    if(              (5)              )
        printf("字符串T（%s）是字符串S（%s）的子序列。\n\n", t, s);
    else
        printf("字符串T（%s）不是字符串S（%s）的子序列。\n\n", t, s);
}
```

Subsequence(s,t)==1