一.单项选择

1. 以下选项中,正确的标识符是(B)。
   A.long
   B._SUM
   c.f(x)
   D.2x

> A.long是关键字，不能作为标识符
> B._SUM
> C.有括号，不合法
> D.数字开头不合法
> 命名规范(第三章PPT 10-12页)
> ·由字母、数字、下划线(Underscore，'_')组成
> ·第一个字符必须为字母或下划线(慎用)
> ·C语言标识符是大小写敏感的(A和a是不同的名字)
> ·用户定义的标识符不应与C语言关键字(Keywords)重复

2. 若变量c为char类型,能正确判断出c为小写字母的表达式是(D)。
   A. 'a'≤c≤'z'
   B.(c≥'a')ll(c≤'z ')
   c. ('a'≤c)and ('z'≥c)
   D.(c≥'a ') &&(c≤'z')

> A.会把'a'≤c的结果(0或1)再与'z'比较，必定是0
> B.恒为1
> C.and不是C语言关键字，要用&&
> D.正确，要同时满足c≥'a'和c≤'z'才为1

3. 若有定义:char str[]="ABCDEF";，则sizeof(str)的值为(D).
   A.4
   B.5
   c. 6
   D. 7

> 6个字符+1个'\0'
> 大小为7

4. 若球体半径定义为:double r;，则求该球体体积的正确表达式为(B)
   A.4/3.0*3.14159*(r^3)
   B.4*3.14159*r*r*r/3
   C.4/3*3.14159*pow(r，3)
   D.4/3*3.14159*r*r*r

> C和D错误之处在于4/3会先计算，两个整型的计算结果为整型，所以4/3=1而不是1.3333
> A中^在C语言中表示位运算异或，而不是指数，指数运算使用math.h的函数pow()计算

5. 若有定义：int a=3,b=2,c=1,z;，则表达式z=a>b>c的值为(A)。
   A.0
   B.1
   C. 2
   D.3

> 3>2为真，所以a>b的值为1
> 1>1为假，所以1>c为假，所以a>b>c为假，所以z=0(真为1，假为0)

6. 逻辑运算符两侧运算对象的数据类型(D)。
   A.只能是0或1
   B.只能是0或非0正数
   C.只能是整型或字符型数据
   D.可以是任何类型的数据

> 任何类型数据都可以，无论整型、字符还是浮点型，0会被视为0(显然)，非0的数都会被视为1

7. 若有定义int a[3][4];则对数组元素不正确的引用是(C)
   A.a[0][2*1]
   B.a[1][3]
   C.a[0][4]
   D.a[4-2][0]

> int a[3][4];
> a[i][j] 0≤i≤2 0≤j≤3为正确的引用

8. 设有语句：int a=2,b=3,c=4; float x=3.5,y=4.8;
   则表达式!(a+b)+c-1&&b+c/2和表达式x+a%3*(int)(x+y)%2/4的值分别为(B)
   A.0和3.50000
   B.1和3.50000
   C.0和4.50000
   D.1和4.50000

> 运算符优先级见第三章PPT 57页

类别	运算符	结合性
后缀	() [] → . ++ - -	从左到右
一元	+ - ! ~ ++ - - (type)* & sizeof	从右到左
乘除	* / %	从左到右
加减	+ -	从左到右
移位	<< >>	从左到右
关系	< ≤ > ≥	从左到右
相等	== ≠	从左到右
位与 AND	&	从左到右
位异或 XOR	^	从左到右
位或 OR	|	从左到右
逻辑与 AND	&&	从左到右
逻辑或 OR	||	从左到右
条件	?:	从右到左
赋值	= += -= *= /= %=>>= <<= &= ^= |=	从右到左
逗号	,	从左到右

```
!(a+b)+c-1&&b+c/2的计算
int a=2,b=3,c=4;
(a+b)=5 c/2=2
!(a+5)=0
!(a+5)+c-1=3 b+c/2=5
!(a+b)+c-1&&b+c/2 = 3 && 5 = 1
```

```
x+a%3(int)(x+y)%2/4
int a=2; float x=3.5,y=4.8;
x+y=8.3
(int)(x+y)=8 (向下取整)
x+a%3 (int)(x+y)%2/4
=3.5 + 2 % 3 * 8 % 2 / 4
=3.5 + 2 * 8 % 2 / 4
=3.5 + 16 % 2 / 4
=3.5 + 0 / 4
=3.5
```

9. C语言程序中使用条件分支语句if~else时，else应与(C)组成配对关系。

A.同一复合语句内部的if

B.在其之前任意的if

C.在其之前未配对的最近的if

D.首行位置相同的if

> if 语句嵌套时，else子句与 if 的匹配原则：与在它上面、距它最近、且尚未匹配的 if 配对，与缩进无关。(第三章PPT 92页)

10. 设有定义int k=0; 则以下k值不是1的是(D)。
A. k++ B. k+=1 C. ++k D. k+1

> A B C都是赋值语句，D仅仅计算，没有对k进行赋值，所以k仍然为0

11. 已知char x[]="hello", y[]={'h','e','l','l','o'}; 则关于两个数组长度的正确描述是(B)。

A.相同

B.x大于y

C.x小于y

D.以上答案都不对

> "hello"是一个字符串，结尾隐藏一个'\0'
> y[] = {'h','e', 'l', 'l', 'o'};只是对字符串数组每个元素初始化，不含有'\0'
> 这里y的定义和int a[]={1,2,3,4,5}类似，都是得到一个五个元素的数组，只是数组y的元素类型为char，a的元素类型为int

12. 以下选项中，操作数必须是整型或字符型的运算符是 (C)。
A. ++

B.!

C.%

D./

> 求余、自增和自减等运算符通常要求操作数是整型类型，自增和自减也可以用于浮点型，但效率远不如整型(第三章PPT 69)

13. 关于C语言程序，以下叙述中正确的是(D)。
A. main函数必须位于所有其他函数之前
B.预处理命令属于一类特殊的C语言语句
C.优先级高的运算符优先计算
D.C语言的输入和输出功能只能通过函数调用才能实现

A.其他函数可以在main函数前定义
B.在C语言的程序中可包括各种以符号#开头的编译指令，这些指令称为预处理命令。预处理命令属于C语言编译器，而不是C语言的组成部分。
C.优先级高的运算符优先结合，而不是优先计算
例如：后缀++和--的优先级最高，但规则是在整个表达式其它所有运算符之后运算 第三章PPT 56页
b = a[i++]中++优先级最高，但是++最后计算
D选项： C语言也没有专门的输入语句，而是用函数接收输入 第二章PPT16页
C语言没有专门的输出语句，而是用函数输出结果 第二章PPT 12页

14. x、i、j、k都是 int型变量，执行表达式x=(i=31, j=25, k=16)后x的值是(D)。
A) 0 B) 31 C) 25 D) 16

> 逗号表达式 见第三章PPT 54页
> 逗号表达式的值取决于最后一个表达式，赋值表达式的值等于赋的值的值，所以(i=31, j=25, k=16)的值即为(k=16)的值，即为16

15. 对二维数组的说明，正确的有几个(C).

```
1. int a[][]={{1,2,3},{4,5,6} };
2. int a[2][]={1,2,3,4,5,6};
3. int a[][3]={1,2,3,4,5,6};
4. int a[2][3]={1,2,3,4,5,6};
```

A) 0 B) 1 C) 2 D) 3

> 第三章PPT 141页
> 每对[]内都必须有一个整型常量表达式(除第一对外…) 所以1和2是错误的，3和4正确

16. 已知: char a; long b; float c;double d;执行语句“a=a+b+c+d;”后，变量 a的数据类型是 (A)
A)char
B) float
C) double
D)以上都不是表达式

> 无论怎么赋值，变量a的类型都不会变，定义时候是什么类型，就是什么类型
> 在a参与计算的过程中可能涉及类型的转换，但是变量a的类型是确定的

17. sizeof("m\x43\\\np\182q")的值为(C)。
A) 16 B) 15 C) 10 D)8

> 第三章PPT 36页转义字符

1. m
2. \x43
3. \ 转义字符\
4. \n 转义字符换行
5. p
6. \1
7. 8
8. 2
9. q
10. \0

共十个字符
\x后最多可以加两位十六进制数表示一个字符
\后最多可以加三个八进制数表示一个字符，但是\1后面是8,八进制中没有数字8，所以\1表示一个字符，8表示另一个字符
同学可以自行printf("%d", sizeof("m\x43\\np\182q"));来测试

18. 关于break语句和continue语句，以下叙述中正确的是 （C）.
A.break语句和continue语句仅可用于循环语句
B.break语句可直接退出多层循环
C. continue 语句提前结束本次循环
D.break语句在退出循环时可携带一个返回值

A.错，比如switch case也可以用break,但这不是循环
B.只能跳出当前循环
C.正确
D.break无此功能

19. 已知学生记录及变量的定义如下

```
struct student {
    int no;
    char name[20];
    char gender;
    struct {int year,month,day;} birth;
}struct student s;
```

以下能给s中的year成员赋值2005的语句是(D)
A) s.year=2005;
B) s[year]=2005;
C) s→year=2005;
D) s.birth.year=2005;

> 结构体的成员是一个结构体，要逐层访问成员 第三章PPT 172页

20. 已知ch是字符型变量，下面不正确的赋值语句是(A)。

A. ch='a+b'

B. ch='\0'

C.ch='7'+'9

D. ch=5+9

> ''内只能有一个字符(转义字符算一个整体)，a+b算三个字符，不能放进''里

21. 执行下列程序后，变量i的值是(B).

```
int i=10,b=1;
switch(i){
    case 9: ++i;
    case 10: i*2;
    case 11: b=(i=++b,i+3,i/3);
          break;
    default : i+=1;
}
```

A)20

B)2

C)11

D)1

> switch case 第三章PPT 97 98 99页
>
> 每个case后面“常量表达式”的值，必须各不相同
>
> 常量表达式仅起语句标号的作用，匹配成功后从此标号开始
>
> 执行，直至遇到break语句才结束
>
> int i=10,b=1;
>
> switch(i){
>
> case 9: ++i; //不执行i不变，i=10
>
> case 10: i*2; //i=20
>
> case 11: b=(i=++b,i+3,i/3); //i=++b后i=2，i+3没有对i赋值所以i不变，i/3同理
>
> break; //跳出switch，i最终为2
>
> default : i+=1;

```
}
```

1. 当运行时输入：abcd$abcde，下面程序的运行结果是(C).

```
#include<stdio.h>
int main(){
    while(putchar(getchar())!='$');
    printf("end”);
}
```

A) abcd$abcde

B)abcdend

C)abcd$end

D)abcd$abcdeend

> 输入$后循环结束，输出end，注意$本身也会输出

23. 以下程序的输出结果是(D)

```
#include<stdio.h>
int main(){
    int a,b;
    for(a=1,b=1:a<=100;a++){
        if(b>=10) break;
        if(b%3==1){
            b+=3;
            continue;
        }
    }
    printf("%d\n”,a);
    return 0;
}
```

A) 101 B)6 C)15 D)4

a=1时
b=1，满足b%3==1,所以b+=3后,b=4。这时我们可以发现每次循环b%3==1都会满足
a=2时 b=7， a=3时 b=10，此时由于先判断b ⩾ 10后执行b+=3，所以在a=3时还没有跳出循环
接下来a=4,判断b ⩾ 0为真，循环结束，a最终为4

24. 以下程序所表示的分段函数是(D).

```c
#include<stdio.h>
int main()
    int x,y;
    printf("Enter x:");
    scanf("%d”,&x);
    y=x>=0?2*x+1:0;
    printf("x=%d:f(x)=%d",x,y);
    return 0;
}
```

A. $f(x)=\begin{cases}0\ (x\leq 0)\\ 2x+1\ (x>0)\end{cases}$

B. $f(x)=\begin{cases}0\ (x\geq 0)\\ 2x+1\ (x<0)\end{cases}$

C. $f(x)=\begin{cases}2x+1\ (x<0)\\ 0\ (x\geq 0)\end{cases}$

D. $f(x)=\begin{cases}0\ (x<0)\\ 2x+1\ (x\geq 0)\end{cases}$

> 显然x⩾0时y=2*x+1，否则为0
> 这里强调一下，C语言中没有≥和≤，要用⩾ ⩽，后面手写代码有同学使用≥和≤，我们进行了一定程度扣分

二、不定项选择题(共9分，每题1.5分)

1. 设x、y和z是int型变量，且x=3,y=4,z=5,则下面表达式中值为0的是(BD).

   A)'x'&&'y'

   B)x⩾y;

   C)x||y+2&&y-z

   D)! ((x<y) && !z || 1)

   > A. 'x'与'y'均非0，逻辑运算非0则视为1，即1&&1=1，&&后仍为1
   > B.3⩾4为假，表达式的值为0
   > C.x非0，直接短路运算，表达式为1 第三章PPT 64页
   > D.(x<y)为1，!z为0， (x<y) && !z为0,0||1为1，最后取反，表达式的值为0

2. 以下叙述中错误的是(BD).

   A)同一个数组中所有元素的类型相同
   B) 初始化时不可以跳过前面的数组元素给后面的元素赋初值
   C) 定义语句int a[10]={0}:给a数组中所有元素赋初值0
   D) 若有定义语句 Int a[4]={1,2,3,4,5}; 编译时将忽略多余的初值

A.数组是存储多个同类型数据的对象 第三章PPT 131页
B.第三章PPT 136页
C99允许指定初始化，将指定的元素赋值，其它为0
例如 int a[6]={[5]=50}; 等价于
int a[6]={0,0,0,0,0,50};
此时赋值的顺序可以任意
例如 int a[15]={ [2]=29, [14]=7, [9]=48};
C. 正确 第三章PPT 134页
D.初值数据项的个数不能多于数组元素的个数，但允许少于数组元素的个数，此时，依次从初值列表中取值对前面的数组元素赋值，而其余的数组元素隐含用0赋初值,初值数据项的个数多于数组元素的个数会报错 第三章PPT 135页

3. 以下表达式的值是整型的有(AD/ACD)
   A: sizeof(double)　　B: 3.5-0.5
   C: 'x'　　D: 3.5>0.5

A.整型，返回对象占据内存的大小
B.浮点型
C.有一定争议，'x'是字符类型，但是某些归类中也把字符类型视为一种特殊的整型
D.为1，整型

4. 以下关于编译预处理的叙述中正确的是(ABD)
   A)预处理命令行必须以#开始
   B)一条有效的预处理命令必须单独占据一行
   C)预处理命令行只能位于源程序中所有语句之前
   D)预处理命令不是C语言本身的组成部分

参考C语言手册https://zh.cppreference.com/w/c/preprocessor

预处理指令控制预处理器的行为。每个指令占据一行，且拥有下列格式:
字符
预处理指令(define、undef、include、if、ifdef、ifndef、else、elif、elifdef、elifndef (C23 起)、endif、line、embed (C23 起)、error、warning (C23 起)、pragma 之一)[1]
实参(取决于指令)
换行符
允许空指令(跟随换行符的 #)，而它无效果。

A.正确
B.正确
C.显然不需要，比如#define N 10可以放在函数中，通常将预处理放在程序开头只是因为这样预处理对整个程序生效
D.C中的预处理命令是由ANSIC统一规定的，但它不是C语言的本身组成部分，不能直接对它们进行编译，因为编译程序无法识别它们。必须对程序进行通常的编译(包括词法和语法分析，代码生成，优化等)之前，先对程序中这些特殊的命令进行“预处理”

5. 下列关于结构体类型和结构体变量的说法中，正确的是(ACD)
   A、“结构体”可将不同数据类型，但相互关联的一组数据，组合成一个有机整体使用。
   B、结构体类型中成员名，不可以与程序中的变量同名。
   C、“结构体类型名”和“数据项”的命名规则，与变量名相同。
   D、相同类型的结构体变量间可以相互赋值。

> B.没有这种要求、、、
> D.第三章 PPT 175页

6. 关于C语言中的switch语句，以下选项中正确的有(AB).
   A. switch语句是一种多分支语句。
   B. switch语句中可以没有default分支。
   C.程序执行到下一个case时，跳出switch语句。
   D.switch后的表达式可以是整型、字符型或浮点型。

> B.正确，就像if也不一定非得后面跟着else
> C.不可以，所以break才经常和switch case在一起 第三章PPT 99页
> D.switch( )中的表达式必须能产生一个可列值，可以是整型、字符型或枚举型 第三章PPT 99页

三、填空(共10分，每空1分)

1. 定义 int a=0, b=0, c=0; 语句 c=2>1?(a=1): (b=2); 执行后， 表达式 a+b+c的值是

> 2

2. 若有以下定义: char c='\X41'; 则变量c中包含的字符个数为

> 1

3. 已知a是一个double 型的正数，写出一个赋值表达式，在它执行后a四舍五入保留两位小数(比如a的值原本是12.666666，执行完该表达式后a的值变为12.670000)

```
a=(double)(int)(a*100+0.5)/100
```

4. 定义char a[]="abcdef"; 则语句printf("%s", &a[2]); 的结果是

> cdef

5. 设有定义 chax x, y; 请写出描述x, y 同时为小写字母或者同时为大写字母的表达式:

```
x⩾'a'&&y⩾'a'&&x⩽'z'&&y⩽'z'||x⩾'A'&&y⩾'A'&&x⩽'Z'&&y⩽'Z'
```

6．程序段：

```
unsigned char x=100, y=200;
do ( x= x+y, y=x-y, x=x-y; ) while (0);
printf ("%d %d\n",x, y);
```

运行后，运行结果为

```
200
100
```

7．程序段：

```
int i,s=0;for(i=1;i<=100:i++) s+=i;
```

运行后，s的值为
5050

8．设：int a[2][3]={1,2,3,4,5},

则：a[1][1] 的值为 5

a[1][2]的值为 0

四.程序填空题

1．在一组有序的数据中查找数据，若找到则输出数据在数组中，否则插入该元素。

```
#include <stdio.h>
#define N 10
int main(){
    int a[N+1] = {升序数列初始化值}, i, j, x, flag = 0;
    scanf("%d", &x);  //输入待查找的数x
    for(i = 0; i < N; i++){
        if(a[i] == x){
            _(1)_ ; break;
        }
        else if( _(2)_ )
            break;
    }
    if(flag == 1) printf("x is in array. \n");
    else if(i <= N){
        //将x插入数组a
        for(_(3)_; j >= i - 1 ; j--)  _(4)_ ;
```

```
        _(5)_ ;
    }
    return 0;
}
// 第一种
(1): flag = 1
(2): a[i] > x
(3): j = N - 2
(4): a[j + 2] = a[j + 1]
(5): a[i] = x

//第二种
(1): flag = 1
(2): a[i] > x
(3): j = N - 1, i++
(4): a[j + 1] = a[j]
(5): a[i - 1] = x

//第三种
(1): flag = 1
(2): a[i] > x && ++i
(3): j = N - 1
(4): a[j + 1] = a[j]
(5): a[i - 1] = x
```

2．完成程序，填上适当的语句，实现功能：将输入的大写字母转换为小写字母、小写字母转换为大写字母、其他字符不变，并最后输出。每空仅写一个表达式或语句。

```
#include <stdio.h>
int main(){
    char c;
    c = getchar();
    switch ( (c >= 'A') + (c > 'Z') + (c >= 'a') + (c > 'z') )
    {
    case 1: _(6)_ ; break;
    case 3: _(7)_ ; break;
    }
    printf("%c", c);
}

(6): c = c - 'A' + 'a' // c += 32
(7): c = c - 'a' + 'A' // c -= 32
```

3．统计字符串中字母的个数，请填空。

```
#include <stdio.h>
int main(){
```

```
    char str[50];
    int i, _(8)_ ;
    scanf("%s", _(9)_ );
    for(i = 0; _(10)_ ; i++)
        if( _(11)_ ) j++;
    printf("j=%d\n", j);
}

(8): j = 0
(9): str
(10): a[i]
(11): str[i] >= 'a' && str[i] <= 'z' || str[i] >= 'A' && str[i] <= 'Z'
```

4. 完成程序，填上适当的语句，实现功能：输入整数n的值，逆序输出n的各位数字。例如：输入3210，输出：0123。每空仅写一个表达式或语句。

```
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", &n);
    do{
        printf("%d", _(12)_ );
    } while( _(13)_ );
    return 0;
}

(12): c % 10
(13): c /= 10
```

5.以下数组a中存放N个由小到大排列的有序整数。把从键盘输入的整数m插入到数组a中，使插入后的数组a仍然有序，请填空。

```
#define N 6
int main(){
    int i, j, m;
    int a[ _(14)_ ] = {10, 20, 30, 40, 50, 60};
    scanf("%d", &m);
    for(j = 0; j < N; j++)
        if( _(15)_ < a[j]) break;
    for(i = N; i > j; i--)
        a[i] = a[ _(16)_ ];
    a[j] = _(17)_;
    for(i = 0; i < N + 1; i++)
        printf("%d", a[i]);
}

(14): N + 1
```

```
(15): m
(16): i - 1
(17): m
```

6. 写一个程序根据收入金额salary(≥ 0)对应不同税率计算应缴税额，并返回应缴税额。税率计算公式f(x)如下:

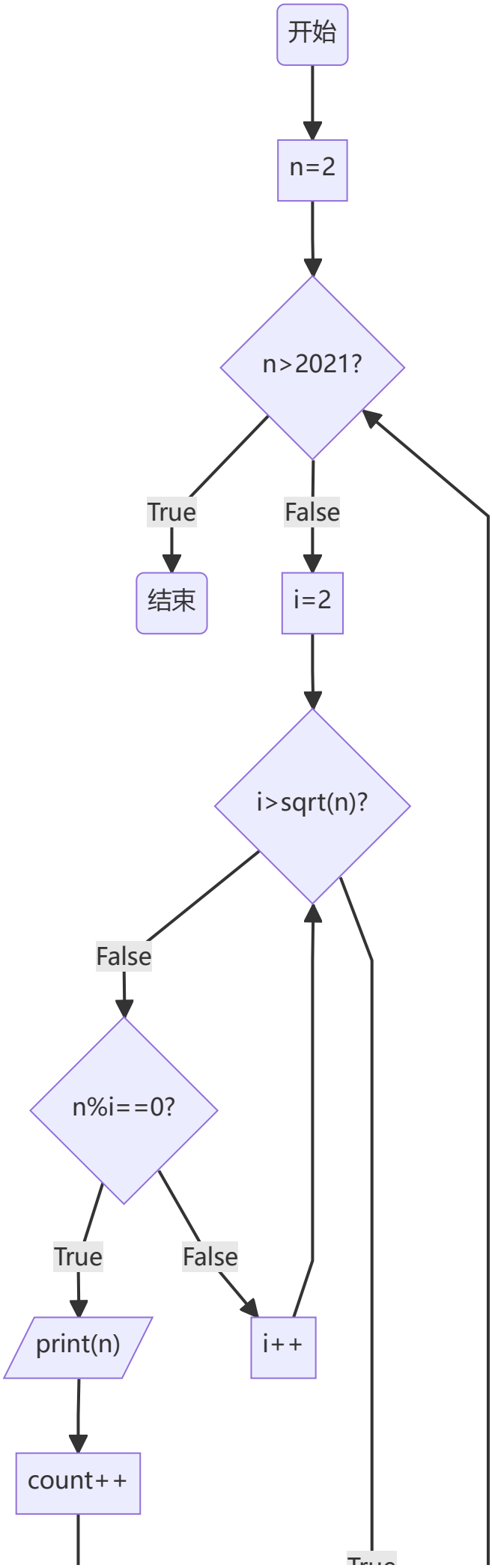
$$f(x)=\begin{cases}0, & x<1000\\ 5\%, & 1000\leq x<3500\\ 10\%, & 3500\leq x<5000\\ 15\%, & x\geq 5000\end{cases}$$

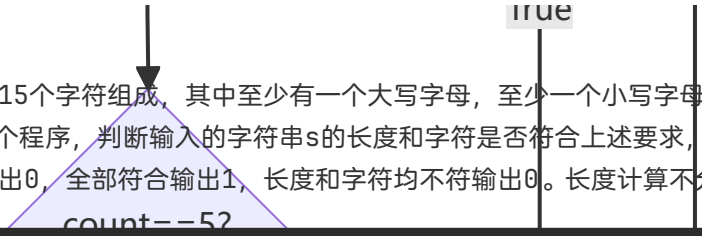
```
int main(){
    int salary;
    scanf("%d", &salary);
    switch ( _(18)_ ){
    case 0: case 1:
        printf("0"); break;
    case 2: case 3: case 4:
    case 5: case 6:
        _(19)_ ;
    case 7: case 8: case 9:
        printf("%f", salary*0.1); break;
    default:
        _(20)_ ;
    };
    return 0;
}

(18): salary / 500
(19): printf("%f", salary * 0.05); break
(20): printf("%f", salary * 0.15); break
```

五.编程题

1. 输出2到2021之间所有的合数(非素数)，每行打印5个合数。不需要编写具体程序，只需要画出对应的流程图即可，流程图要符合规范。


开始
n=2
n>2021?
True
False
结束
i=2
i>sqrt(n)?
False
n%i==0?
True
False
print(n)
i++
count++



2. 规定一串密码由8-15个字符组成，其中至少有一个大写字母，至少一个小写字母，至少一个0-9的数字字符，编写一个程序，判断输入的字符串s的长度和字符是否符合上述要求，长度不符输出-1，字符不符输出0，全部符合输出1，长度和字符均不符输出0。长度计算不允许调用库函数。

```c
#include <stdio.h>
#include <stdbool.h>

#define MAX 100
#define CHAR_LEGAL (upch && lwch && num)
int main(){
    char password[MAX]; //尽量开大一些 不要是16这种
    gets(password);
    bool upch = false, lwch = false, num = false;
    int len = 0;
    for (int i = 0; password[i]; i++, len++){
        if(password[i] >= 'a' && password[i] <= 'z') lwch = true;
        else if(password[i] >= 'A' && password[i] <= 'Z') upch = true;
        else if(password[i] >= '0' && password[i] <= '9') num = true;
    }
    if (!CHAR_LEGAL) putchar('0');
    else if (len < 8 || len > 15) puts("-1");
    else putchar('0');
}
```

3. 编写一个程序，通过键盘输入双精度浮点数数组a(数组中所有元素值不重复且**已单调增排序**)，返回最接近且**小于等于**所有元素平均值的元素值。如：a[5]={0.1,0.2,0.3,0.4,0.5} 返回0.3。

```c
/*
编写一个程序，通过键盘输入双精度浮点数数组a(数组中所有元素值不重复且已单调增排序)，返回最接近且小于等于所有元素平均值的元素值。如：a[5]={0.1,0.2,0.3,0.4,0.5} 返回0.3。
*/
#include <stdio.h>
#define N 100 //数组最大长度
int main(){ //注意main函数的返回值必须是int型，不能是double error: '::main' must return 'int'
    int n,sum,average;
    scanf("%d",&n); //注意输入整型使用%d，scanf的参数是地址，需要&
    double a[N]; //尽可能不要使用double a[n]，在一些编译器无法通过编译，定义数组是定义一个长度为常数的数组
    for(int i=0;i<n;i++){ //注意从0到i<n，或i<=n-1；如果数组大小比n大，也可以i=1到i<=n，但要和sum处一致
        scanf("%lf",&a[i]); //输入double型%lf，a是数组地址，a[i]是数组的元素而不是地址，所以需要&取地址
    }
```

```
    //数组只能通过循环来一个一个输入，不能使用一个scanf输入整个数组(字符串可以scanf("%s",str);来输入，str为字符串地址)
    sum = 0; //注意初始化sum
    for(int i=0;i<n;i++){
        sum += a[i];
    }
    average = sum / n;
    int min = 0;
    for(int i=0;i<n;i++){
        if(a[i] > average){ //注意不要让数组下标越界，因为数组是严格升序，一定存在一个元素a[i]>average
            //或if(a[i]<=average && a[i+1]>average) 输出a[i]
            printf("%f\n",a[i-1]); //一般使用%f输出double和float，%lf也算对
            break; //注意break，及时结束循环
        }
    }

    return 0;
}
```

4. 写一个程序，从键盘输入一个5x5的矩阵。
   1)实现将矩阵转置处理。注：矩阵转置在数学上的定义为：设A为mxn阶矩阵(即m行n列的矩阵)，其第i行第j列的元素是a(i.j)。定义A的转置为这样一个nxm阶矩阵B，满足B的第i行第j列元素是A的第j行第i列元素，即b(ij)=a(ji)。
   2)将该矩阵与其转置矩阵相乘，输出相乘后的结果，要求分行打印。

```
#include <stdio.h>
int main()
{
    int i, j, k;
    int A[5][5];
    int AT[5][5];
    // 输入
    for(i = 0; i< 5; i++)
    {
        for ( j = 0; j < 5; j++)
        {
            scanf("%d", &A[i][j]);
            AT[j][i] = A[i][j];   // 坐标交换赋值(合并为一步也可)
        }
    }

    // 转置
    /*for(i = 0; i< 5; i++)
    {
        for ( j = 0; j < 5; j++)
```

```
        {
            AT[j][i] = A[i][j];    // 坐标交换赋值
        }
    }*/

    /*
    // 输出检验转置
    for ( i = 0; i < 5; i++)
    {
        for ( j = 0; j < 5; j++)
        {
            printf("%d ", AT[i][j]);
        }
        printf("\n");
    }*/

    // 乘法
    int mul[5][5] = {0};
    for(i = 0; i < 5; i++)
    {
        for ( j = 0; j < 5; j++)
        {
            for (k = 0; k < 5; k++)
            {
                mul[i][j] += (A[i][k] * A[k][j]);
            }
        }
    }

    // 输出结果
    for ( i = 0; i < 5; i++)
    {
        for ( j = 0; j < 5; j++)
        {
            printf("%5d", mul[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```