سوال 8:

قسمت اول:

قسمت هایی که اشتباه اند با قرمز و قسمت هایی که درست اند با سبز مشخص شده اند:

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

مشکل این قسمت آن است که main باحرف بزرگ آمده است.

```
// int c = 10;   ---------------------------->      int c=10;
```

از آنجایی که می خواهیم از متغیر C در ادامه استفاده کنیم پس باید از قبل تعریف کرده باشیم و لذا نباید جمله بالا کامنت باشد.

```
printf("c is now %d", c);
int input;
scanf("%d", input); ---------------------->      scanf("%d",& input);
```

در scanf برای گرفتن عدد از ورودی باید قبل از اسم متغیر & بیاید.البته موقع کامپایل اروری نمی دهد و بعد از اجرا به ارور برمیخوریم.

```
double 2xResult = input * 2;---------->      double xResult=input *2;
```

نباید اسم متغیر با عدد شروع شود.

```
printf("%lf", 2xResult); ------------------>      printf("%lf", xResult);
```

مطابق قبلی اینجا نیزمتغیر نباید با عدد شروع شود.

```
return "HelloWorld";
```

مقداری که return برمیگرداند یک عدد است و البته با وجود HelloWorld مشکلی در اجرای برنامه به وجود نمی آید ولی warning می دهد و دقیقا HelloWorld را برنمیگرداند.

```
}
```

قسمت دوم:

```
int main( ) {
int a = 0, b;
int i, j, k, l;
char x;
double c, d;
```

a -= -5 - 5; ----------→ a=10

b = -3 - - (- 3) ; ----------→ b=-6

ابتدا منفی سه داخل پرانتز در یک منفی پشت پرانتز ضرب میشود و میشود سه ی مثبت حال منفی سه را منهای سه میکنیم.

c = a + 7; ----------→ c=17.000000

در اینجا a که یک مقدار int هست با 7 جمع شده و سپس یک cast اتفاق افتاده و جواب در c که به فرم دابل است ذخیره شده.

d = b + 4.0; ----------→ d=-2.000000

در اینجا نیز مقدار صحیح b با یک cast به فرم دابل برده شده است.

x = a + b + 65; ----------→ x=10+(-6)+65=69 → x=E

در سمت راست معادله پس از انجام عملیات های جمع یک عدد صحیح داریم. از آنجایی که در طرف چپx یک کاراکتر است پس آن مقدار صحیح کد اسکی کاراکتر می باشد و با یک cast به فرم کاراکتر تبدیل میشود.

i = j = k = l = 1;

i *= (k += (2 * (l -= (3 / j--))));→1-i*=(k += (2 * (l -= (3))))→2-i*=(k += (2 * (-2)))→3-i*= (k += (-4))→4-i=-3

در مرحله اول سه تقسیم بر j شده سپس از j یکواحد کم میشود و در j ذخیره میشود(j=0). در مرحله دواز l سه واحد کم میشود.(l=-2) در مرحله سوم k با منفی چهار جمع شده و میشود منفی سه.(k=-3) در آخر هم i را در منفی سه ضرب کرده و در خودش ذخیره میکنیم.(i=-3).

printf ("c_int = %d,c_double = %lf, c = %lf\n", (int)c, c, a + 7) ;

در اینجا در قسمت اول c که به فرم دابل بوده رو با cast کردن به فرم int برده و فقط قسمت صحیح عدد نشان داده میشود در قسمت وسط خود c که یک مقدار دابل دارد نمایش داده میشود. در قسمت سوم باید توجه کنیم که a یک int است و با 7 هم جمع شود یک مقدار int می دهد حال که بدون cast می خواهیم مقدار آن را باlf% که برای دابل است نشان دهیم به مقصود خود نمیرسیم و جواب 0.000000 برای ما چاپ میشود.

printf ("d_int = %d,d_double = %lf, d = %lf\n", (int)d, d, b + 4.0) ;

در اینجا نیز در قسمت اول d که به فرم دابل بوده رو با cast کردن به فرم int برده و فقط قسمت صحیح عدد نشان داده میشود در قسمت وسط خود d که یک مقدار دابل دارد نمایش داده میشود. در قسمت سوم b یک مقدار int دارد ولی وقتی با4.0 جمع میشود جواب ما یک عدد اعشاری میشود که حتی بدون cast کردن هم میتوان باlf% که برای دابل است حاصل را نمایش داد.

printf ("x = %c\n", x) ;

با توجه به cast صورت گرفته در بالا خروجی این printf همان کاراکتر E می باشد

printf ("i = %d, j = %d, k =%d, l = %d\n", i, j, k, l);

خروجی های printf فوق در بالا حساب شد.

return 0;

}

```
c_int = 17,c_double = 17.000000, c = 0.000000
d_int = -2,d_double = -2.000000, d = -2.000000
x = E
i = -3, j = 0, k =-3, l = -2
```

خروجی: