مهدی رحمانی 9731701

جواب سوال 1 سری 4:

قسمت اول:

خروجی تابع به صورت زیر می باشد:

25 s

5 r

C adds wings to your thoughts

1

2

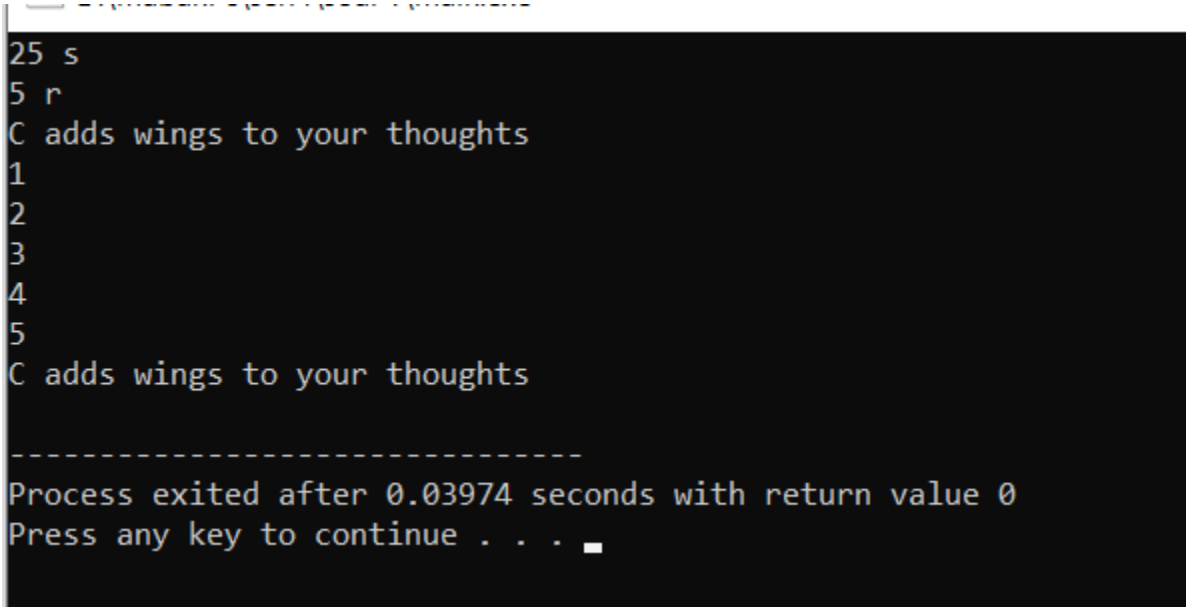
3

4

5

C adds wings to your thoughts

یا اگربخواهم به صورت بهتر بیان کنم به صورت زیر می باشد:



حال عملکرد هرتابع را توضیح می دهیم:

تابع print1:

این تابع نه ورودی دارد و چون void می باشد مقداری را برنمیگرداند. هر بار که این تابع در main فراخوانی می شود ابتدا مقدار i را برابر 0 گذاشته و سپس آن را یک واحد زیاد میکند و1 می شود. سپس شرط آن درهربار فراخوانی برقرار است و وارد آن می شود و یک بار عبارت C adds wings to your thoughts را در خروجی چاپ میکند و بخاطر \n میرود سطر بعد وسپس به عبارت return میرسد و از تابع خارج می شود و عملا عبارت ()print1 که پس از return نوشته شده به کار نمی آید.

تابع print2:

این تابع نیز نه ورودی دارد و چون void می باشد مقداری را برنمیگرداند. زمانی که این تابع فراخوانی میشود ابتدا مقدار متغیر i را استاتیک تعریف میکند (یعنی مقدار i در آن ذخیره می شود و پس از اینکه دوباره تابع فراخوانی شد i همان مقدار نهایی را دارد و دوباره مقدار دهی نمیشود.) ودر بار اول فراخوانی مقدار اولیه آنرا 0 می دهد. سپس i یک واحد زیاد می شود و برابر 1 میشود و شرط چک میشود. در بار اول که تابع فراخوانی شود i در شرط برقرار است و وارد آن میشود و سپس مقدار i درخروجی چاپ میشود و به خاطر \n می رود سطر بعد. و سپس دوباره تابع print2داخل خود تابع فراخوانی میشود. حال چون i به صورت اساتیک بود این بار مقدار اولیه i برابر 1 است وسپس یک واحد زیاد میشود و دوباره مانند قبل در شرط چک میشود. این عمل تازمانی که i<6 میباشد انجام میشود ولی پس از اینکه i>5 شد وارد دستور else شده و به return می رسد و از تابع خارج میشود.

تابع junk :

این تابع دو ورودی دارد که متغیر هایa به صورت int و c به صورت char می باشد. وقتی این تابع در main فراخوانی میشود و ورودی های عددصحیح و کاراکتر به همین ترتیب به آن داده میشود ; این تابع ابتدا آن عدد صحیح را درخودش ضرب میکند و مقدار به دست آمده را با مقدار اولیه آن جایگزین میکند. در مرحله بعد کاراکتری که در ورودی گرفته را با 1 جمع میکند یعنی درواقع یک cast رخ می دهد و کد اسکی آن کاراکتر که عددی صحیح است با یک جمع شده و کد اسکی حرف بعدی به دست می آیدو حال دوباره یک cast روی میدهد و کد به دست آمده به کاراکتر تبدیل شده و با کاراکتر اولی جایگزین میشود. در نهایت به ترتیب مقدار صحیح به دست آمده و کاراکتر به دست آمده یا یک فاصله از هم در خروجی چاپ شده و به خاطر \n به خط بعد میرود.

تابع main :(در این قسمت علت چاپ خروجی ها هم توضیح داده شده)

این تابع که تابع اصلی برنامه است و همه چیز از آنجا شروع میشود ; ابتدا درآن متغیر انتیجر i تعریف شده و مقدار اولیه 5 گرفته(درواقع به این نکته دقت داشت که i هم در print1 و هم درprint2 وهم در main تعریفشده ولی این متغیر چون داخل 3 scope مجزا تعریف شده پس کاری به هم ندارن) و همچنین متغیر از جنس کاراکترc تعریف شده و کاراکتر r درآن ریخته شده.

درابتدا تابع junk فراخوانی شده و ورودی های آن هم همان i ومتغیرc تعریف شده در main می باشند. طبق توضیحات بالا مقدار i که برابر5 است در خودش ضرب شده ومیشود25 و کاراکتر بعد از r که s میاشد در خروجی به ترتیب در سطر اول چاپ میشوند.

در مرحله بعد به دستور print میرسیم و مقدار متغیرهای i و c که در ابتدای main تعریف شده بودند و برابر 5 و r بود با یک فاصله از هم در خروجی چاپ میشوند(در حقیقت تابع junk روی مقادیراصلی این متغیرها تاثیری نمیگذارد و آن ها گلوبال تعریف نشده بودند و متغلق به این scope اند.)

در مرحله بعد تابعprint1 فراخوانی شده و در خط بعد در خروجی عبارت C adds wings to your thoughts چاپ میشود.

در مرحله بعد تابع print2 فراخوانی شده و در خط های بعد به ترتیب مقادیر 1 و2 و3 و4 و 5 (هرکدام دریک خط) چاپ میشوند.

در مرحله بعد هم دوباره تابع print1 فراخوانی شده و در خط آخر عبارت C adds wings to your thoughts چاپ میشود و سپس برنامه تمام میشود.

قسمت دوم:

```
#include <stdio.h>

int check (int m);
```

باید توابعی که در پایین تر از main تعریف شده اند را یک بار در پیش اعلان قبل ازmain به صورت بالا آوریم.(یا به عنوان راه دیگر می توانستیم کلا تابع check زا به طور کامل قبل از main تعریف کنیم که نیازی به پیش اعلان نباشد)

```
main() {
   int k = 35, z;
   z = check (k);
   printf ("\n%d", z);
}
check (m ) {  ----------→ chek( int m){
```

باید تایپ ورودی حتما تعیین شود که با توجه به نوع ورودی که در main داده شده میفهمیم int بوده.(لازم به ذکر است وقتی تایپ خروجی مشخص نشود به صورت اتوماتیک نوع خروجی int در نظر گرفته میشود.

```
   int m;----------  →باید حذف شود
```

وقتی تایپ و متغیر را در ورودی تعریف میکنیم دیگر نیازی نیست یک بار دیگر در تابع تعریف کنیم و خطاست.

```
   if (m > 40)
      return (1);
   else
      return (0);
}
```

البته لازم به ذکر است مثلا اگر درمقابل returnها اگر اعداد داخل () نوشته نمیشدند هم برنامه درست بو د ولی باز با این حال خطا محسوب نمی شود. در ضمن در پایان main خوب بود که یک retun 0 مثلا تعریف میکردیم تا ببینیم آیا برنامه به درستی به انتها رسیده یا خیر. چون تایپ خروجی main را نیز مشخص نکردیم به صورت اتوماتیک یک مقدار اینتیجر برمیگرداند ولی مقدار آن مشخص نیست.