মজার C

Variables
Datatype

WRITTEN BY

MD. SHARIF CHOWDHURY

ভেরিয়েবল ও ডাটা টাইপ

আমরা যখন একটা প্রোগ্রাম লিখি তখন প্রোগ্রামের হিসাবের কাজটা শুধু কম্পিউটার করে । আমাদের কাজ হয় হিসাবটা কিভাবে সুষ্ঠুভাবে কম্পিউটার করবে তা নির্ধারণ করা ।

একটা সাধারন সমস্যাঃ

> *রহিম কে দেওয়া হল ৫ টা আম । করিম কে দেওয়া হল ৭ টা আম । রহিম ও করিমকে মোট কয়টা আম দেওয়া হল ।*

সমাধানঃ

খুব সোজা উত্তর ১২ টা । কিন্তু উত্তর টা কেমন করে এলো । আবারো সোজা উত্তর যোগ করে ।

মূল হিসাব টা হলঃ

রহিম কে দেওয়া হল = ৫ টা আম

করিম কে দেওয়া হল = ৭ টা আম

রহিম ও করিমকে দেওয়া হল = ১২ (৫+৭) টা আম

যোগ মানে হল একসাথে করা । আমরা যদি বাস্তবে এই হিসাবটা করি তাহলে এইরকম হবে । রহিম ৫ টা আম একটা ঝুড়িতে রাখল । করিমও আরেকটা

ঝুড়িতে ৭ টা আম রাখল । তারপর তারা দুই ঝুড়ির আম আরেকটা ঝুড়িতে রাখল । তারপর সব আম হিসাব করা হল ।

এবার আমরা পুরো পদ্ধতিটি লিখে ফেলি ।

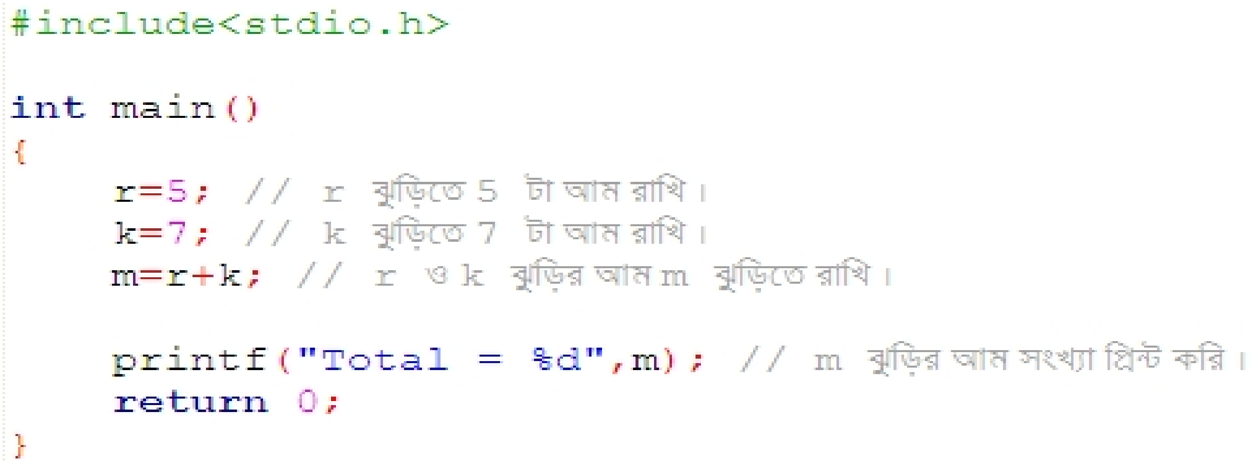
- রহিমের আম “র” নামক ঝুড়িতে রাখি ।
- করিমের আম “ক” নামক ঝুড়িতে রাখি ।
- করিম ও রহিমের আম “ম” নামক ঝুড়িতে রাখি ।
- “ম” এর আম সংখ্যা হিসাব করি ।

“ম” এর আম সংখ্যাই হল মোট আম সংখ্যা ।

এবার এই সমস্যার জন্য একটা প্রোগ্রাম লিখে ফেলি । তার আগে একটা জিনিস শিখি ।

- `x=5;` এর মানে হল x এর মান 5 হয়ে গেল।
- `y=x;` এর মানে হল x এর মান y হয়ে গেল। মানে x এর মান **y** এর মধ্যে চলে আসলো । এখানে x এর মান 5 তাই y এর মান 5 হয়ে গেল ।
- `y=x+y;` এখানে y এর মান হবে x আর y এর মানের সমষ্টির সমান । যেহেতু `x=5; y=5;`তাই `y=x+y=5+5=10;`
- তাহলে এই জিনিসটা স্পষ্ট যে সমীকরণের বাম পাশের চলকের মান হবে ডান পাশের চলক বা চলক সমষ্টির হিসাবকৃত মানের সমান ।

এবার মূল প্রোগ্রাম টা লিখি ।

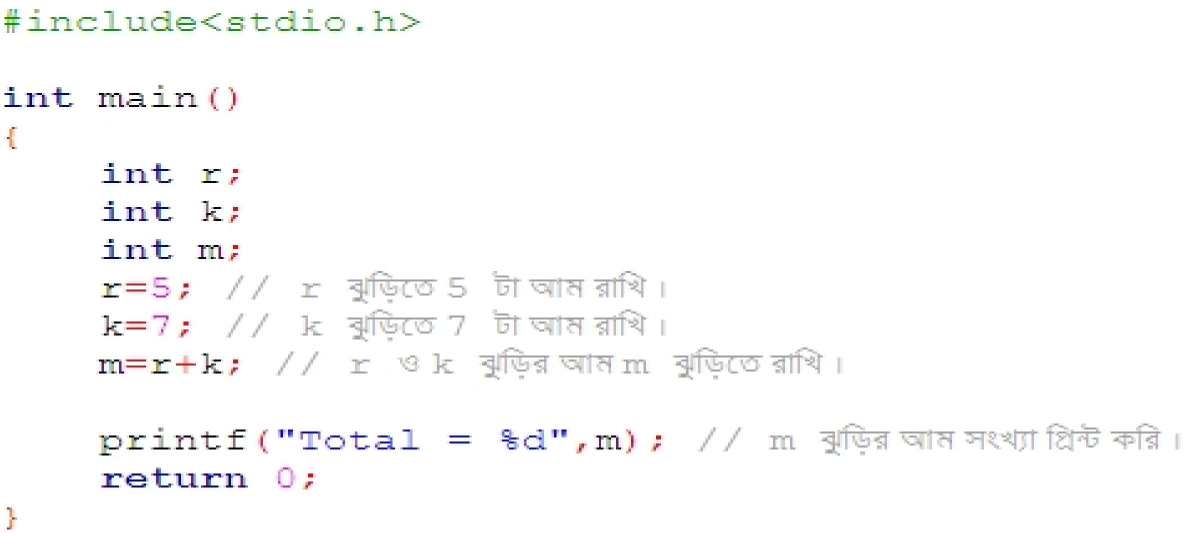


```
#include<stdio.h>

int main()
{
     r=5;  //  r  ঝুড়িতে 5  টা আম রাখি ।
     k=7;  //  k  ঝুড়িতে 7  টা আম রাখি ।
     m=r+k;  //  r  ও k  ঝুড়ির আম m  ঝুড়িতে রাখি ।

     printf("Total = %d",m);  //  m  ঝুড়ির আম সংখ্যা প্রিন্ট করি ।
     return 0;
}
```

printf("Total = %d",m); এই লাইন একটু সমস্যা করলেও বুঝতে অসুবিধা হওয়ার কথা না যে **%d** এর জায়গায় **m** এর মান প্রিন্ট করবে । **%d** কেন ব্যাবহার করা হইছে তা একটু পরে আলোচনা করা হবে ।

এবার প্রোগ্রামটি কম্পাইল ও রান কর । এররর পাবে।

এবার প্রোগ্রামটিকে নিচের মত করে লিখ ।



```
#include<stdio.h>

int main()
{
     int r;
     int k;
     int m;
     r=5;  //  r  ঝুড়িতে 5  টা আম রাখি ।
     k=7;  //  k  ঝুড়িতে 7  টা আম রাখি ।
     m=r+k;  //  r  ও k  ঝুড়ির আম m  ঝুড়িতে রাখি ।

     printf("Total = %d",m);  //  m  ঝুড়ির আম সংখ্যা প্রিন্ট করি ।
     return 0;
}
```

```
Total = 12
```

মাথার মধ্যে অনেকগুলো প্রশ্ন গিজগিজ করতেছে তাই না ??? প্রশ্নগুলো তাড়াতাড়ি খাতার মধ্যে লিখে ফেলো । তারপর পড়া শুরু কর ।

আমরা আমাদের কাজের ধাপগুলো লক্ষ্য করি । আমরা কি সব ধাপগুলো লিখেছি? চিন্তা কর তো কি কি ধাপ লিখি নাই ? ৫ মিনিট চিন্তা কর , তারপর আবার পড়া শুরু কর ।

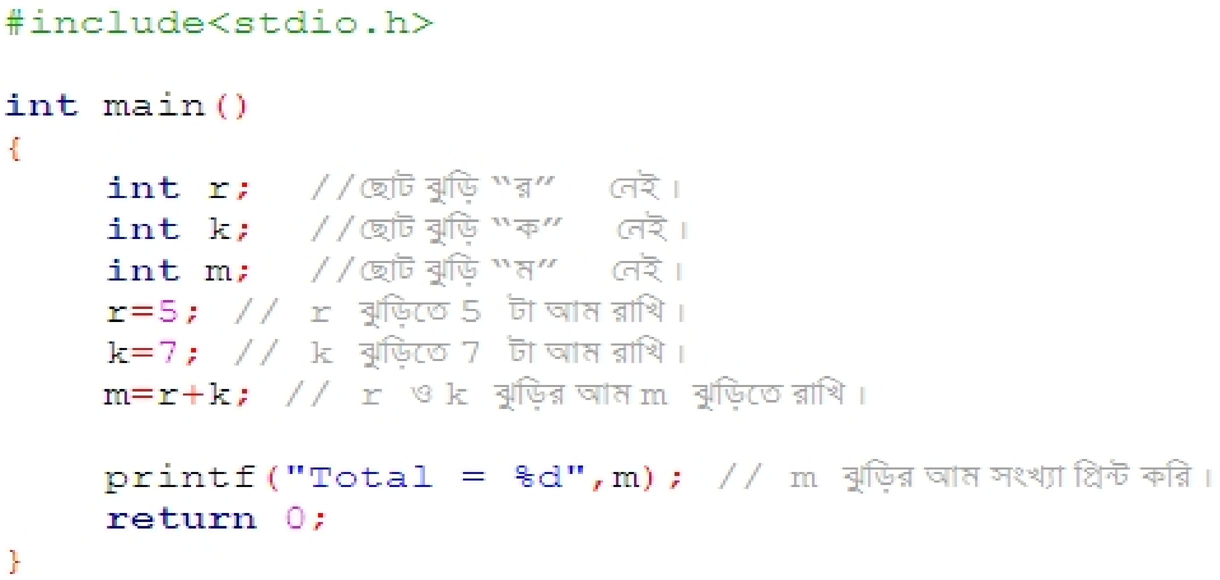
আচ্ছা ধাপগুলো এইরকম হওয়া উচিত কি না ...।

- "র" নামক ঝুড়ি নেই ।
- "ক" নামক ঝুড়ি নেই ।
- "ম" নামক ঝুড়ি নেই ।
- রহিমের আম "র" নামক ঝুড়িতে রাখি ।
- করিমের আম "ক" নামক ঝুড়িতে রাখি ।
- করিম ও রহিমের আম "ম" নামক ঝুড়িতে রাখি ।
- "ম" এর আম সংখ্যা হিসাব করি ।

এবার শেষ হয় নাই । আচ্ছা রহিম যে ঝুড়িটা নিবে তা ছোট না বড় কি রকম হবে । কারন ৫ টা আম রাখতে ছোট ঝুড়ি দরকার , কিন্তু ৫০ আম রাখতে দরকার বড় ঝুড়ি । তাহলে এইবার পুরো ধাপগুলো মিলায় দেখ ।

- ছোট ঝুড়ি “র” নেই।
- ছোট ঝুড়ি “ক” নেই।
- ছোট ঝুড়ি “ম” নেই।
- রহিমের আম “র” নামক ঝুড়িতে রাখি।
- করিমের আম “ক” নামক ঝুড়িতে রাখি।
- করিম ও রহিমের আম “ম” নামক ঝুড়িতে রাখি।
- “ম” এর আম সংখ্যা হিসাব করি।

এবার নিচের প্রোগ্রামটার সাথে মিলায় নাও।



```
#include<stdio.h>

int main()
{
    int r;    //ছোট ঝুড়ি “র”    নেই।
    int k;    //ছোট ঝুড়ি “ক”    নেই।
    int m;    //ছোট ঝুড়ি “ম”    নেই।
    r=5;  //  r  ঝুড়িতে 5  টা আম রাখি।
    k=7;  //  k  ঝুড়িতে 7  টা আম রাখি।
    m=r+k;  //  r  ও k  ঝুড়ির আম m  ঝুড়িতে রাখি।

    printf("Total = %d",m);  //  m  ঝুড়ির আম সংখ্যা প্রিন্ট করি।
    return 0;
}
```

int কি?

int হল ডাটাটাইপ। যেমন গল্পে ঝুড়ির টাইপ হল ছোট। আমরা প্রোগ্রামে অনেক ডাটা ব্যাবহার করি। কোন ডাটা ব্যাবহার করার আগে তার ডাটাটাইপ অবশ্যই বলে দিতে হবে।

ভেরিয়েবল

আচ্ছা তোমার মনে কি এই প্রশ্নটা কখনো আসছে যে আমরা এখানে r,k,m নিলাম কেন? এবার r এর জায়গার rohim লিখে দেখত কি হয় ? তারপর রহিম এর জায়গায় rohim1 , rohim_1 , rohim 1 , rohim-1, 1rohim, _rohim , _1rohim লিখে দেখ আর কম্পাইল করে রান কর। লক্ষ্য করবা কিছু কিছু ক্ষেত্রে এররর দেখাচ্ছে। আর কিছু কিছু ক্ষেত্রে ঠিক আছে।

প্রোগ্রামিং এর ভাষায় এই r , k , m কে ভেরিয়েবল বলে। ভেরিয়েবল হল প্রোগ্রাম এর সেই উপাদান যার মধ্যে বিভিন্ন মান রাখা হয়। যেমনঃ ঝুড়ি। ঝুড়ির মূল কাজটা যেমন আম রাখা তেমনি ভেরিয়েবল এর কাজ হল বিভিন্ন উপাদান এর মান রাখা। এই ভেরিয়েবল লিখার কিছু নিয়ম আছে।

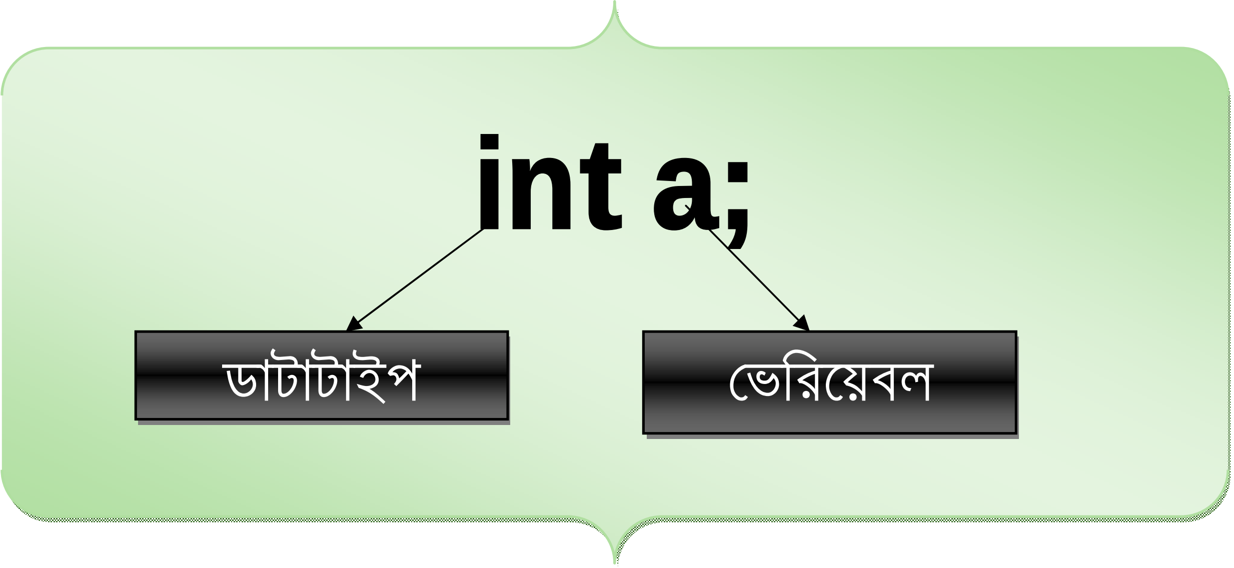
- ভেরিয়েবল এর নাম অবশ্যই a-z , A-Z , 0-9 , “_”(আন্ডারস্কর) এর মধ্যে হতে হবে। যেমনঃ abcd , abcFD, ab_cd .
- নামের শুরু হবে a-z , A-Z বা “_” দিয়ে। যেমনঃ _abcd,Abcd,aBcd.
- নাম ৩২ অক্ষর এর বেশি হবে না। বেশি হলে পরের গুলো হিসাবে ধরা হবে না।
- নাম কোন keyword হওয়া যাবে না।

Keyword

সি তে এমন কিছু word আছে যাদের কাজ নির্দিষ্ট ও সীমিত । এদেরকে বলে keyword . সি তে মোট ৩২ টি keyword আছে । গুগল এ সার্চ দাও পেয়ে যাবা ।

[তোমরা কি কখনো গুগল এ এইকথাগুলো লিখে সার্চ দিয়েছ? tips google search]

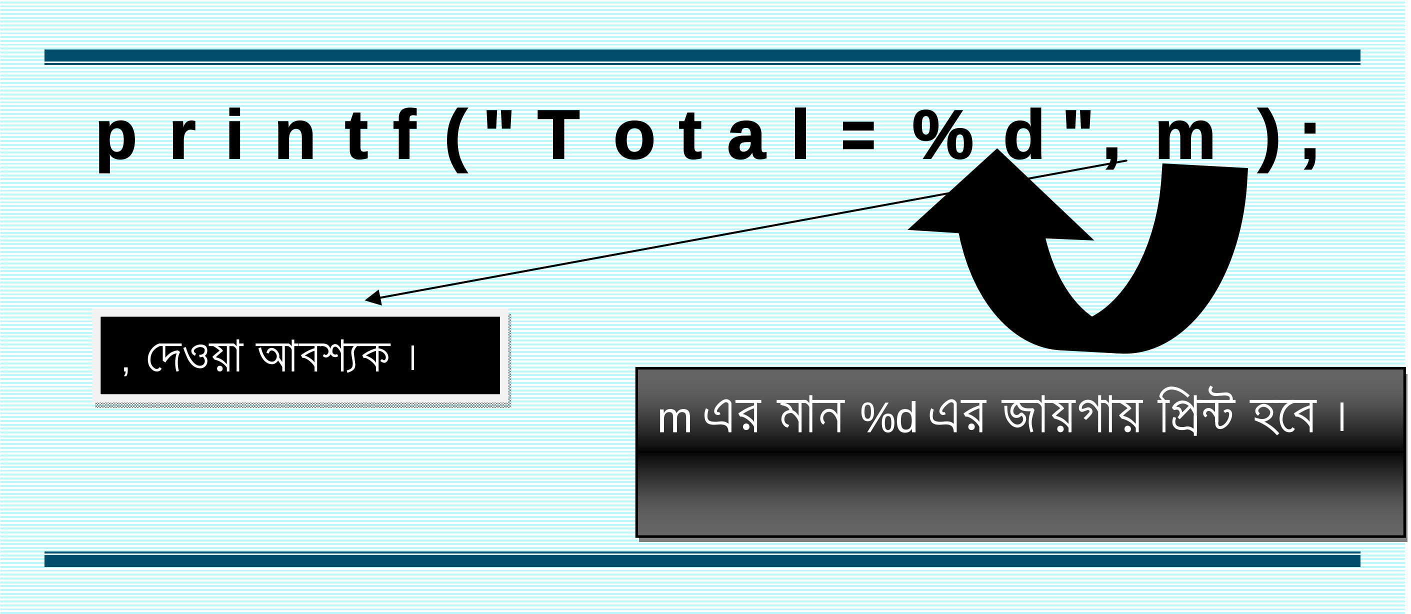
লক্ষ্য করে দেখ বেশ কয়েকটা keyword তোমরা পেয়ে গেছো । প্রোগ্রাম করার জন্য এই keyword গুলো মুখস্ত করার কোন দরকার নাই । ধিরে ধিরে এমনিতেই মুখস্ত হয়ে যাবে । মানে প্রোগ্রাম করতে করতে আয়ত্ত হয়ে যাবে। codeblocks এ keyword গুলো আলাদা রঙ (সাধারনত নীল) এর হয় ।



এইভাবে কোন ভেরিয়েবল ব্যাবহার করার আগে তার ডাটাটাইপ নির্ধারণ করে দিতে হয় । একে ভেরিয়েবল ডিক্লিয়ার করা বলে ।

মোটামুটি আলোচনা করা শেষ।

এবার কিছু বিষয় লক্ষ্য করা যাক ।

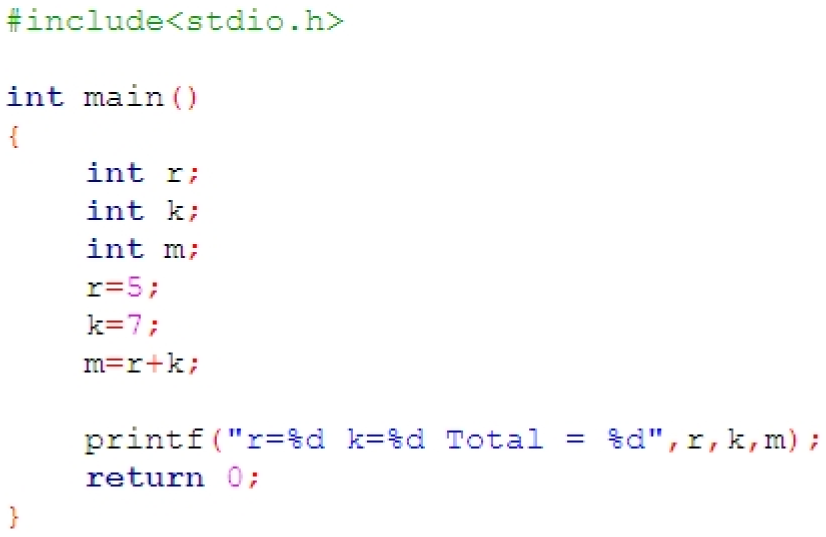


`printf("Total=%d", m);`

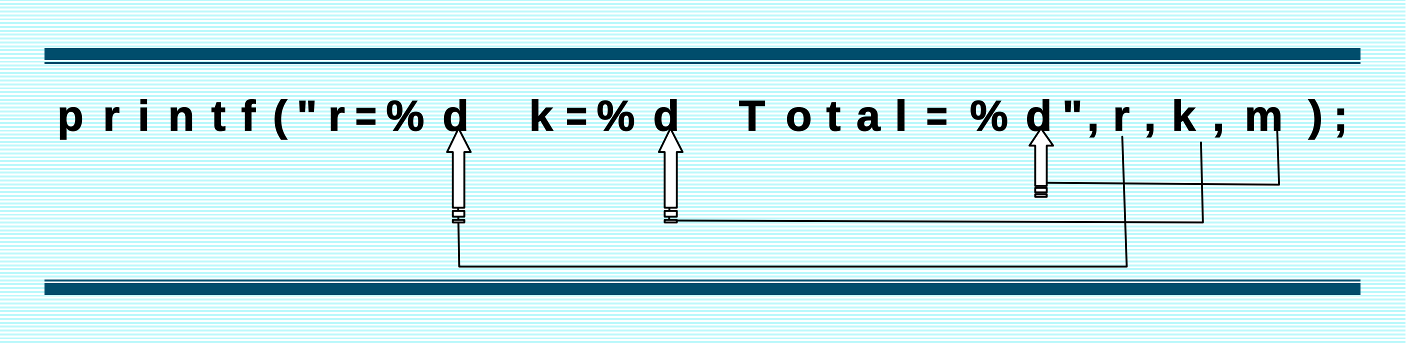
, দেওয়া আবশ্যক ।

m এর মান %d এর জায়গায় প্রিন্ট হবে ।

এবার নিচের প্রোগ্রামটির আউটপুট (যা প্রিন্ট করে দেখায়) কতো হবে বলোতো ।

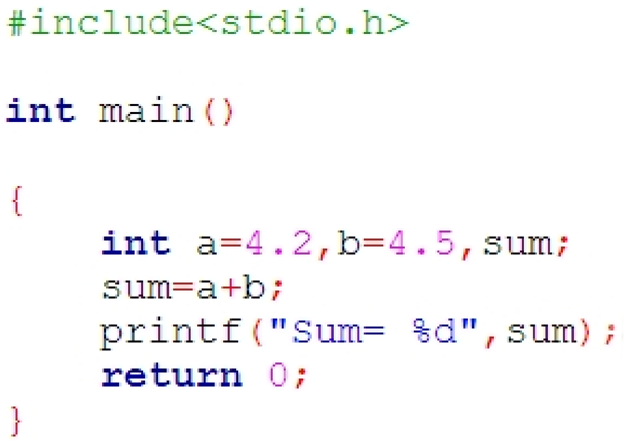


```
#include<stdio.h>

int main()
{
    int r;
    int k;
    int m;
    r=5;
    k=7;
    m=r+k;

    printf("r=%d k=%d Total = %d",r,k,m);
    return 0;
}
```



```
printf("r=%d  k=%d  Total=%d",r,k,m);
```

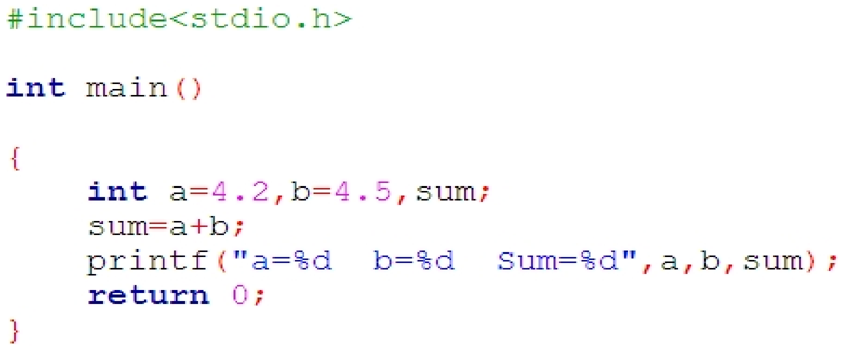
এবার নিচের প্রোগ্রামটি দেখ । বলতো আউটপুট কতো হবে??



```
#include<stdio.h>

int main()

{
     int a=4.2,b=4.5,sum;
     sum=a+b;
     printf("Sum= %d",sum);
     return 0;
}
```

হিসাব মতে এখানে sum এর মান 4.7 হওয়ার কথা । এবার কম্পাইল করে রান করো । দেখবে অউটপুট হচ্ছে 8। কারন বুঝতে পার নাই?????????

তাহলে এবার নিচের প্রোগ্রামটি লেখ ।



```
#include<stdio.h>

int main()

{
     int a=4.2,b=4.5,sum;
     sum=a+b;
     printf("a=%d  b=%d  Sum=%d",a,b,sum);
     return 0;
}
```

কম্পাইল ও রান করো। এবং ভালভাবে লক্ষ্য করো।

এইটাতো বুঝতে পেরেছ যে a ও b এর মান 4 এইজন্য sum এর মান 8 হইছে। কিন্তু কেন a ও b এর মান 4 হইছে?

int

int হল পূর্ণ সংখ্যা। যেমন 4,5,9,100,2,9, এতে দশমিক এর পর কোন কিছু হিসাবে ধরা হয় না।

দশমিক এর পরে হিসাবে আনতে গেলে ব্যাবহার করতে হয় float, double I

এবার নিচের প্রোগ্রামটা দেখ।

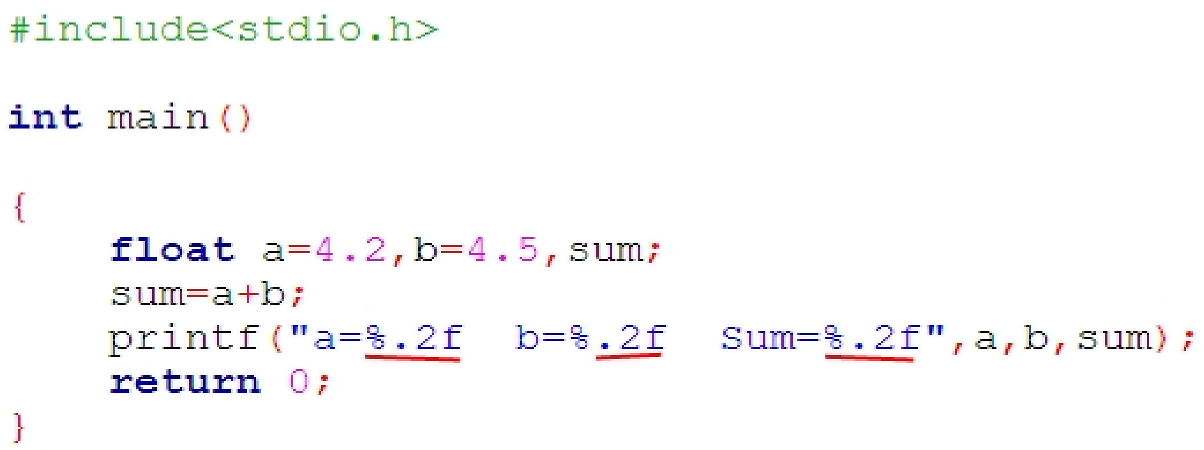
```
#include<stdio.h>

int main()

{
    float a=4.2,b=4.5,sum;
    sum=a+b;
    printf("a=%f  b=%f  Sum=%f",a,b,sum); // float এর জন্য %f
    return 0;
}
```

এর আউটপুট

```
a=4.200000  b=4.500000  Sum=8.700000
```

এতগুলা ০ দেখতে কেমন জানি লাগতেছে।

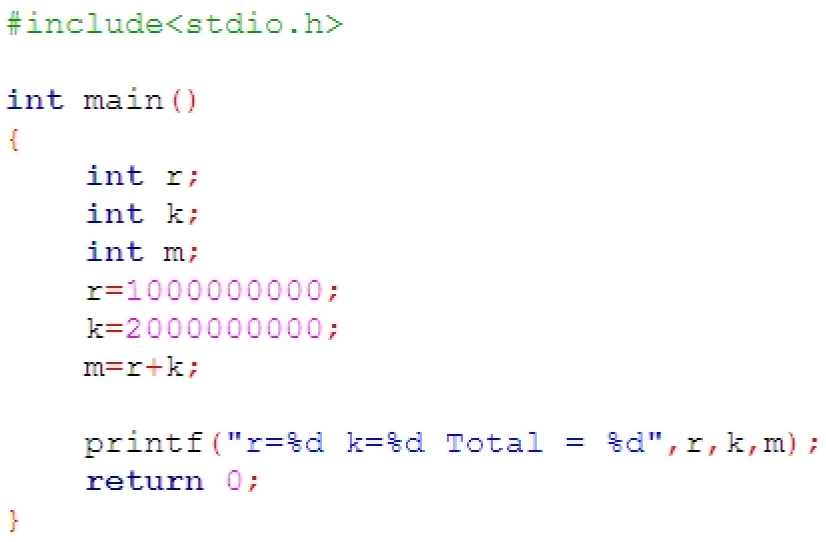
এবার নিচের প্রোগ্রামটা লেখ, কম্পাইল করে রান করো। তারপর লক্ষ্য করো।



```
#include<stdio.h>

int main()

{
      float a=4.2,b=4.5,sum;
      sum=a+b;
      printf("a=%.2f  b=%.2f  Sum=%.2f",a,b,sum);
      return 0;
}
```

এরপরেও বুঝতে না পারলে %2f এর জায়গায় %3f, %4f দিয়ে দেখ বুঝতে পারবে ।

এবার নিচের প্রোগ্রামটা দেখ, দেখে বলো কি প্রিন্ট করবে । যদি তোমার বুঝতে সমস্যা হয় কি প্রিন্ট করবে তাহলে তুমি ঠিকমতো প্রাকটিস করো নি । আর যদি বুঝতে সমস্যা না হয় তাহলে লিখে ফেলো আউটপুট । তারপর কম্পাইল করে রান করো । দেখ তুমি সঠিক কি না !!!!!!!!!!!!!



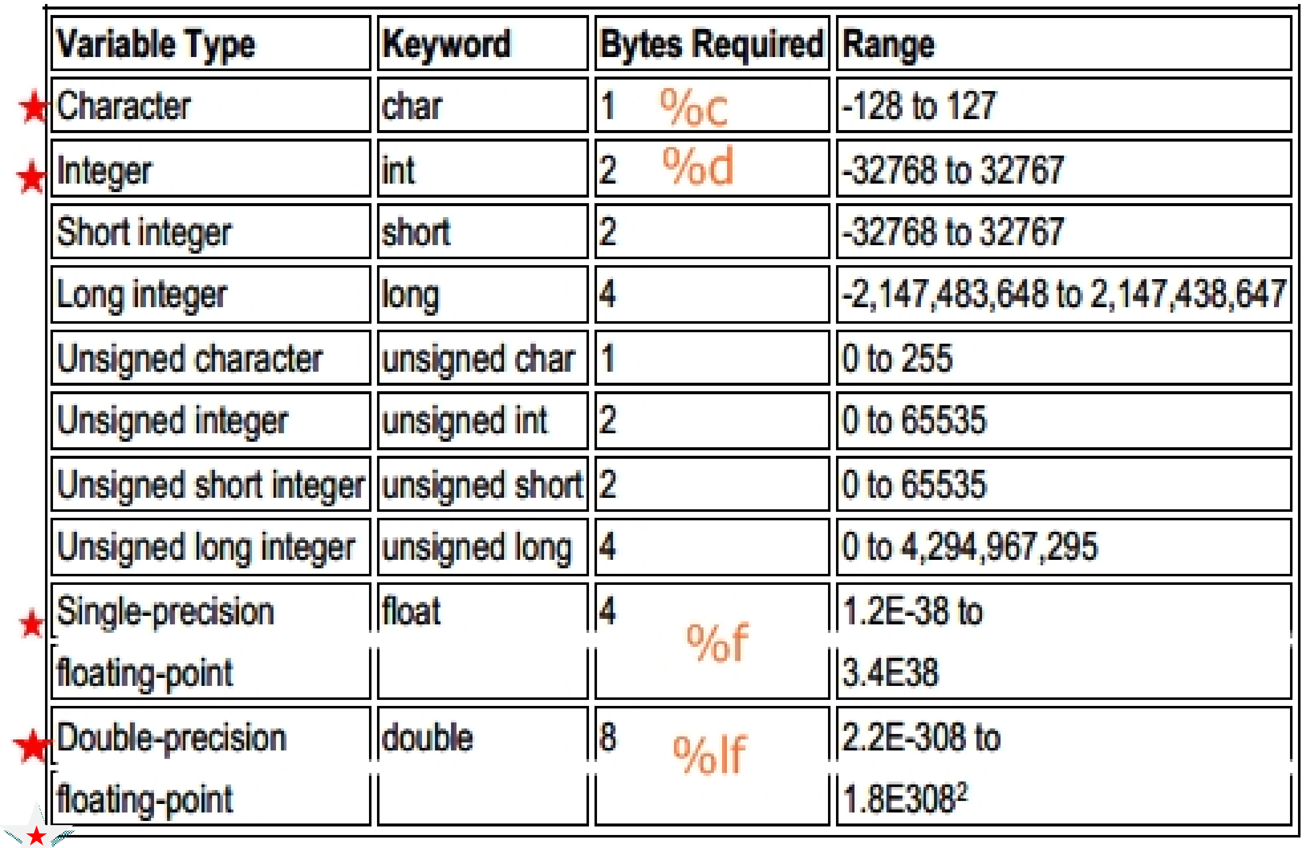
```
#include<stdio.h>

int main()
{
     int r;
     int k;
     int m;
     r=1000000000;
     k=2000000000;
     m=r+k;

     printf("r=%d k=%d Total = %d",r,k,m);
     return 0;
}
```

r , k এর মান ঠিক আসলেও m মানে মোট মানটা ভুল আসছে । কিন্তু কেন ?

চিন্তা করো ! চিন্তা করো ! না পারলে পরা শুরু করো পারলে মিলায় নাও ।

ধর র , ক , ম তিনটা ঝুড়ির ধারন ক্ষমতা ১০ টা আম । তাহলে র আর ক ঝুড়িতে যথাক্রমে ৫ ও ৭ টা আম রাখা কোন সমস্যাই না । কিন্তু যখন এই ৫ আর ৭ টা আম ম ঝুড়িতে রাখা হবে তখন সমস্যা । মোট আম ১২ টা আর ঝুড়ির ধারন ক্ষমতা ১০ টা । হিসাব তো গোলমাল হবেই । এর জন্য ম ঝুড়িটা এমন হতে হবে যাতে কমপক্ষে ১২ টা আম ঝুড়িতে ধরে ।

এবার জেনে নেই বিভিন্ন রেঞ্জের ডাটা টাইপ ।



Variable Type	Keyword	Bytes Required	Range
★ Character	char	1 %c	-128 to 127
★ Integer	int	2 %d	-32768 to 32767
Short integer	short	2	-32768 to 32767
Long integer	long	4	-2,147,483,648 to 2,147,438,647
Unsigned character	unsigned char	1	0 to 255
Unsigned integer	unsigned int	2	0 to 65535
Unsigned short integer	unsigned short	2	0 to 65535
Unsigned long integer	unsigned long	4	0 to 4,294,967,295
★ Single-precision floating-point	float	4 %f	1.2E-38 to 3.4E38
★ Double-precision floating-point	double	8 %lf	2.2E-308 to 1.8E308[2]

চিহ্ন গুলো জানা জরুরী । printf এ এগুলা কিভাবে ব্যাবহার করবে তা হয়ত বুঝতে পারছ । না পারলে টেবিলের Bytes Required এ দেখ বুঝতে পারবা ।

তবে codeblocks এ int এর রেঞ্জ টেবিলের থেকে বেশি ।

Md. Sharif Chowdhury

HAJEE MOHAMMAD DANESH SCIENCE & TECHNOLOGY UNIVERSITY

DINAJPUR

sharif.cse.hstu@gmail.com (**facebook**)

shariftech.wordpress.com