|  |
| --- |
| **امیررضا ویشته 99522221**  **99521424احمدرضا طهماسبی** |
| Project1 09/09/1401  *Operation System* |

سیستم کال sort کردن پراسس هادر سیستم عامل XV6 :

برای تعریف این سیستم کال در xv6 باید فایل های زیر را تغییر دهیم:

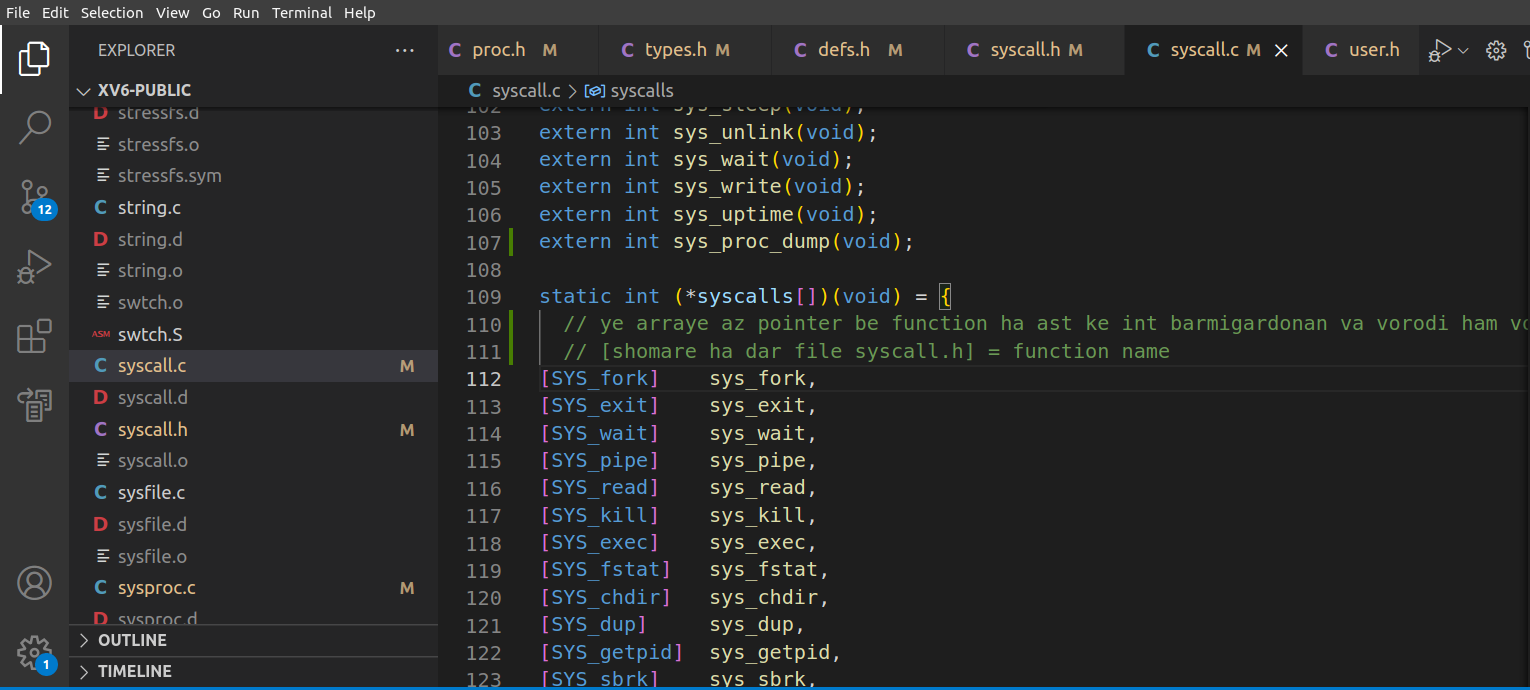
1. syscall.h
2. syscall.c
3. user.h
4. defs.h
5. sysproc.c
6. proc.c
7. makefile
8. ps.c

درابتدا باید سیتم کال خود را در فایل syscall.h تعریف میکنیم و به ان یک عدد نسبت میدهیم زیرا در سطح سیستم عامل همه چیز بر حسب عدد است

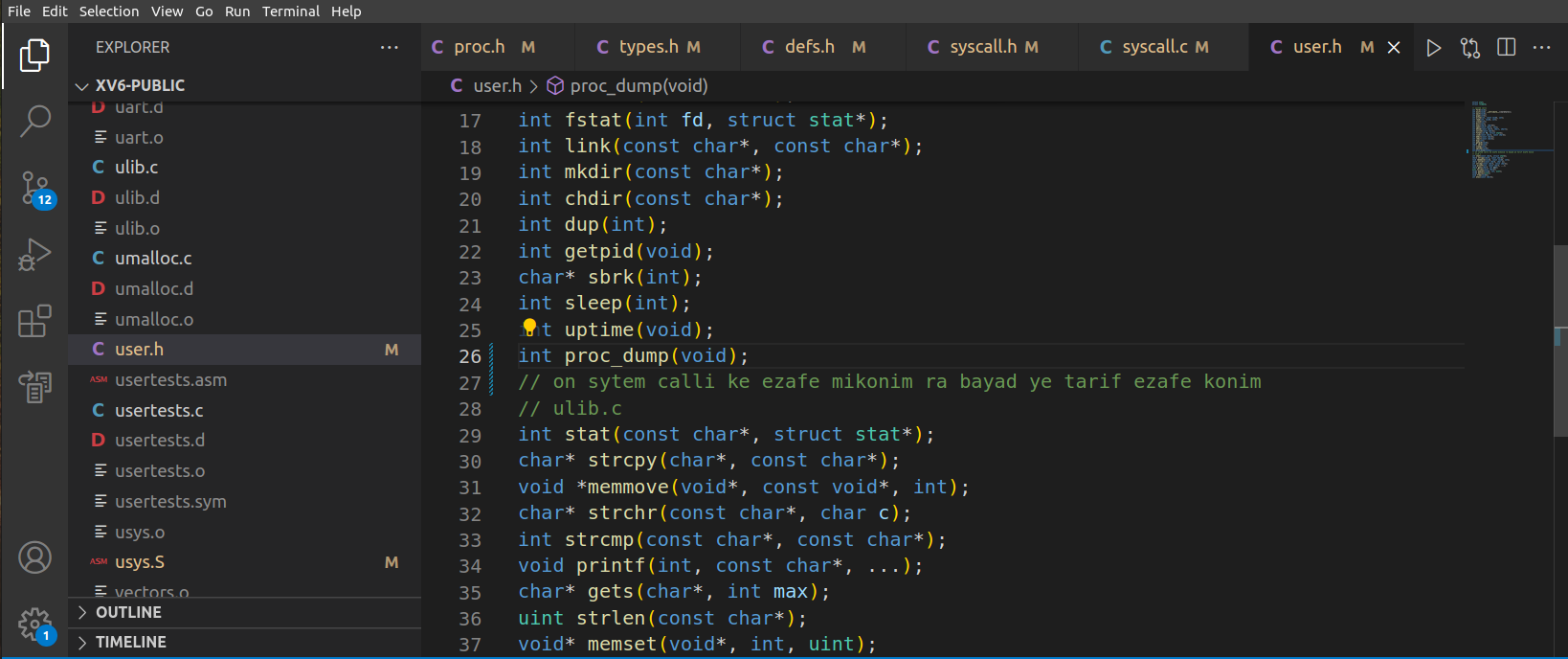


سپس در فایل syscall.c یک امضا از تابع میگذاریم

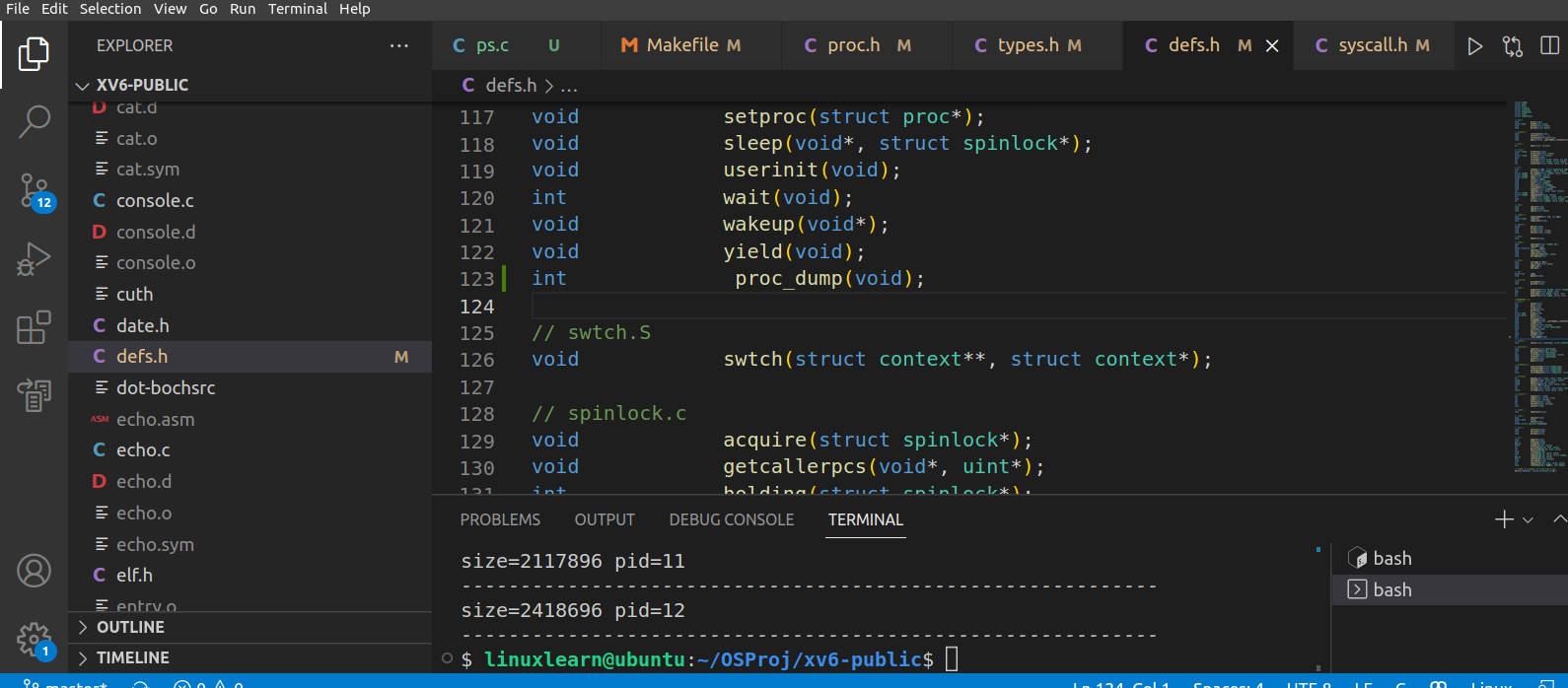




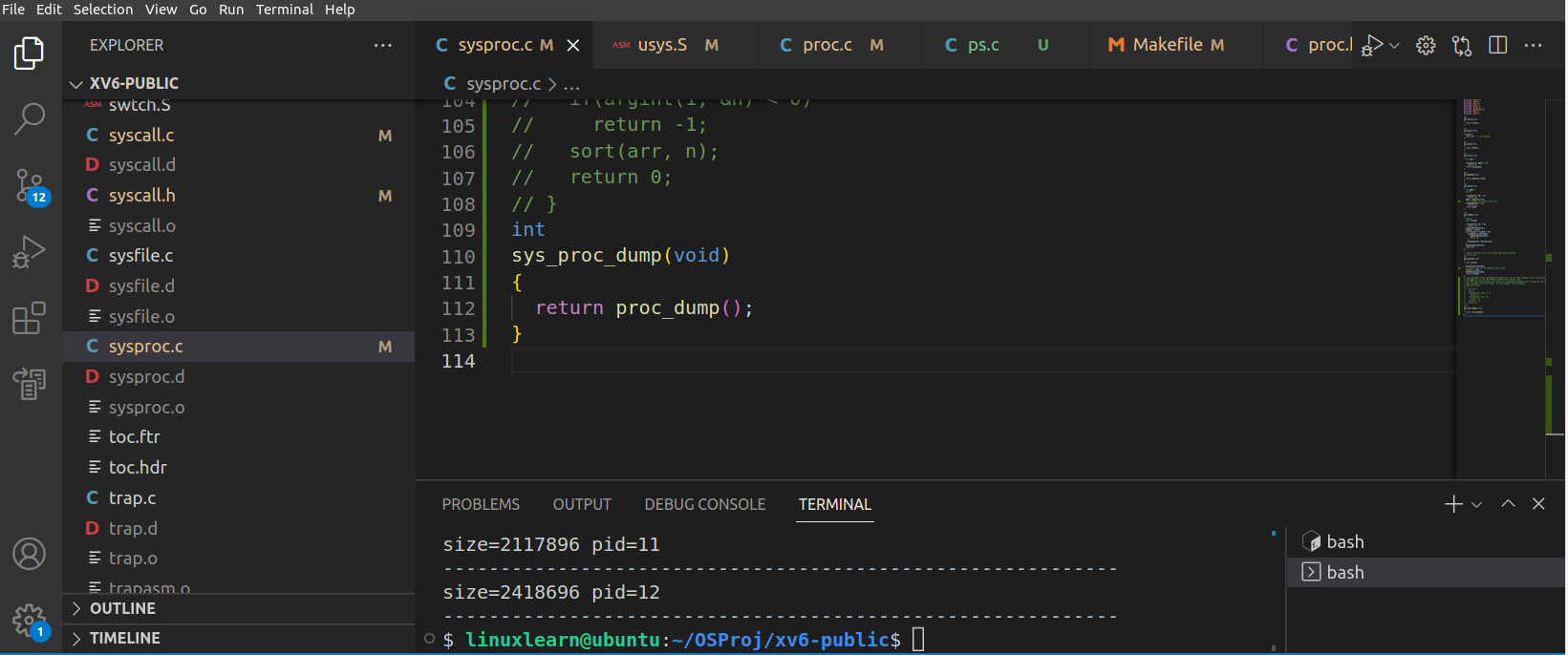
به این فایل user.h نیز مشابه قبلی فقط امضایی از تابع شامل نحوه ورودی و خروجی اضافه میشود



فایل defs.h نیز مثل دو مورد قبل عمل میکنیم:

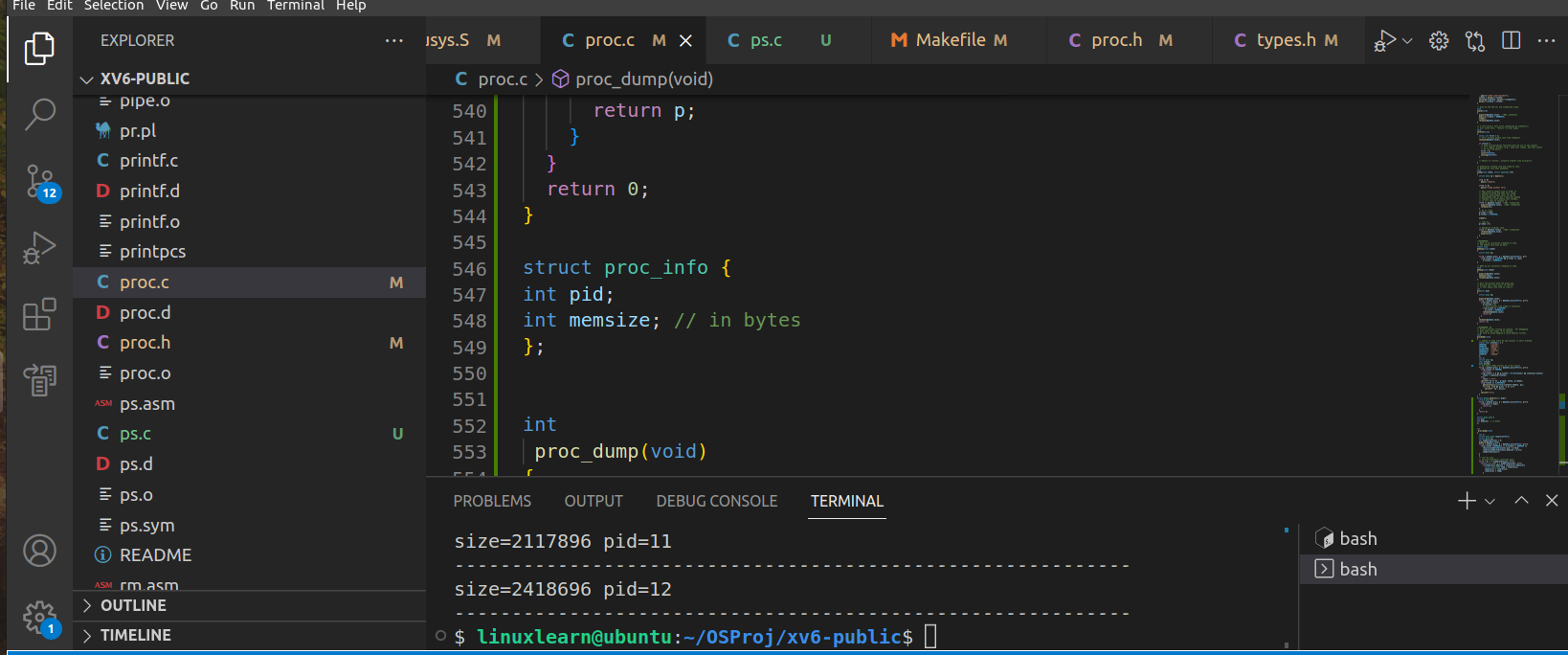


فایل sysproc.c در این فایل ما صرفا خروجی تابع proc\_dump نوشته شده در فایل proc.c را برمیگردانیم



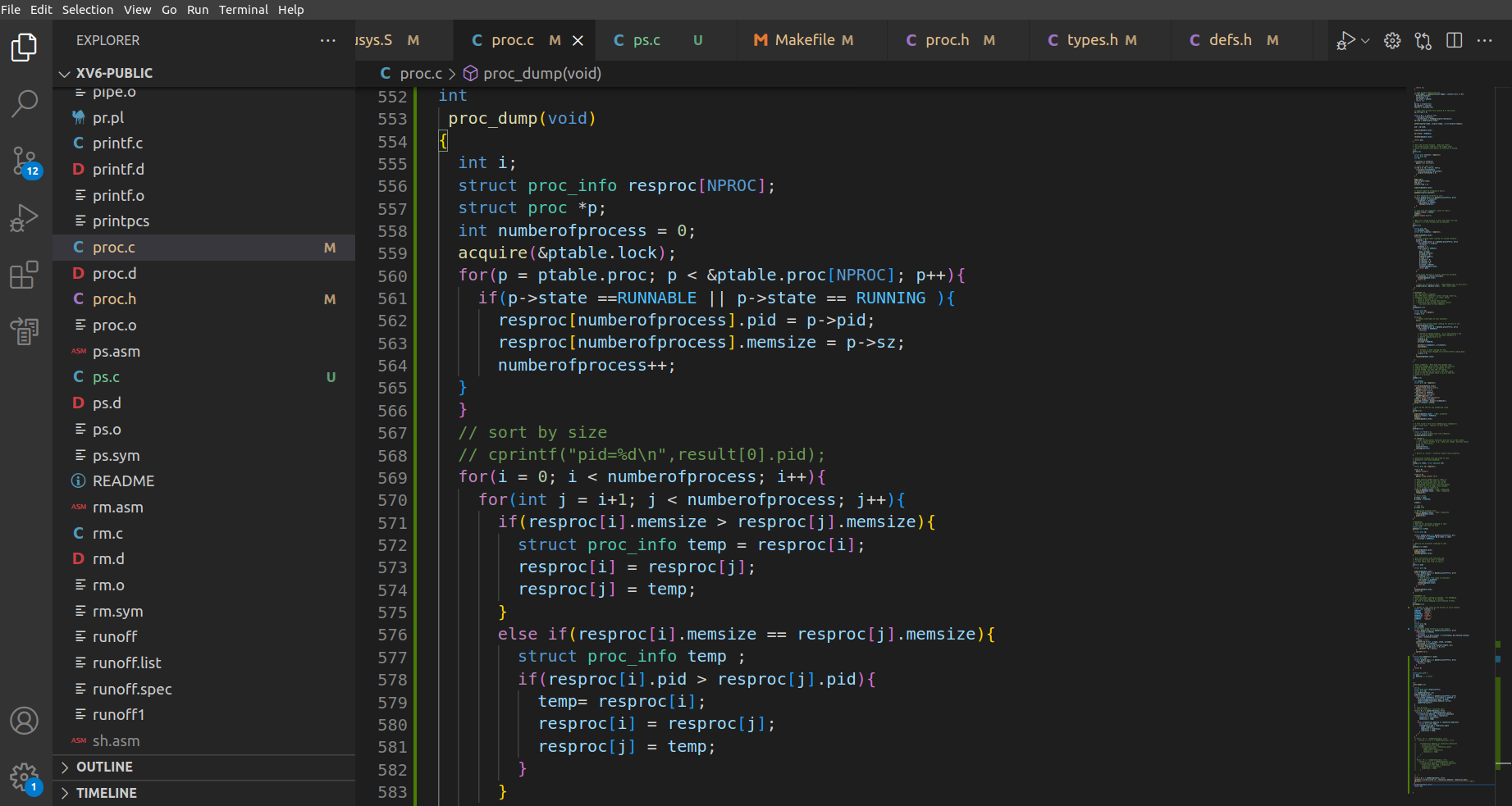
در فایل proc.c ما تابع اصلی که process ها را sort میکند مینویسیم که در واقع با استفاده از ptable همه پراسس ها را میگیرد و ابتدا انها را در یک لیست به نام resproc که از نوعproc\_info است میریزدبه خاطر اینکه این struct جدید داده شده دارای حجم کتری از ویژگی های یک پراسس است در واقع تنها ویژگی هایی را دارد که برای مرتب کردن به انها نیاز داریم

Lock هم در بخش مرتب کردن پراسس ها زده ایم تا چند پراسس با هم وارد ciritical section نشوند

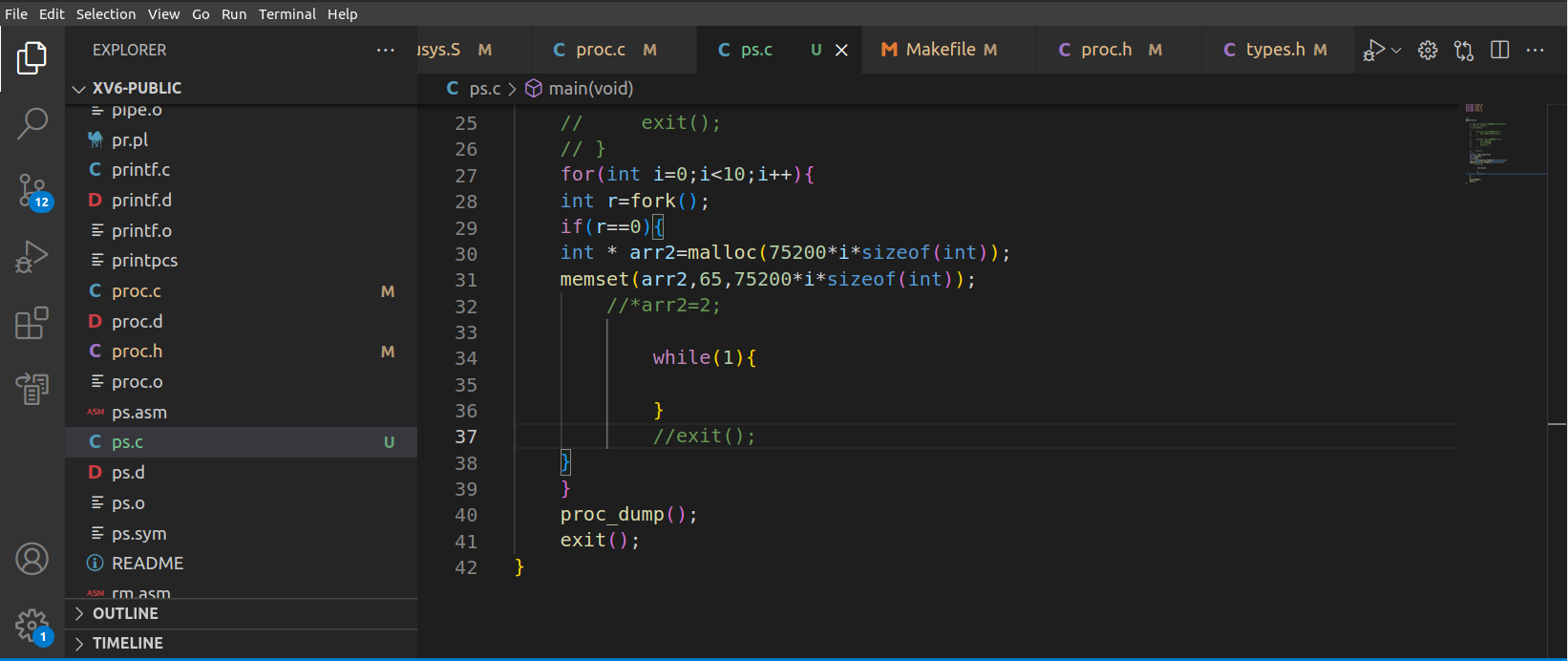


من این struct proc\_info را در همین فایل proc.c تعریف کرده ام

سپس دو for این تابع را مرتب کرده ام به این شکل که ابتدا سایز process ها را ملاک قرار داده ام و بعد pid را .



سپس در فایل ps.c تست این کد را به این شکل نوشته ام که یک تعداد process با fork() ایجاد کرده ام و بعد با دستور malloc سایز این process ها را تغییرداده ام و به تابع proc\_dump() برای مرتب کردن داده ام دستور memset() هم صرفا برای پر کردن این ارایه است

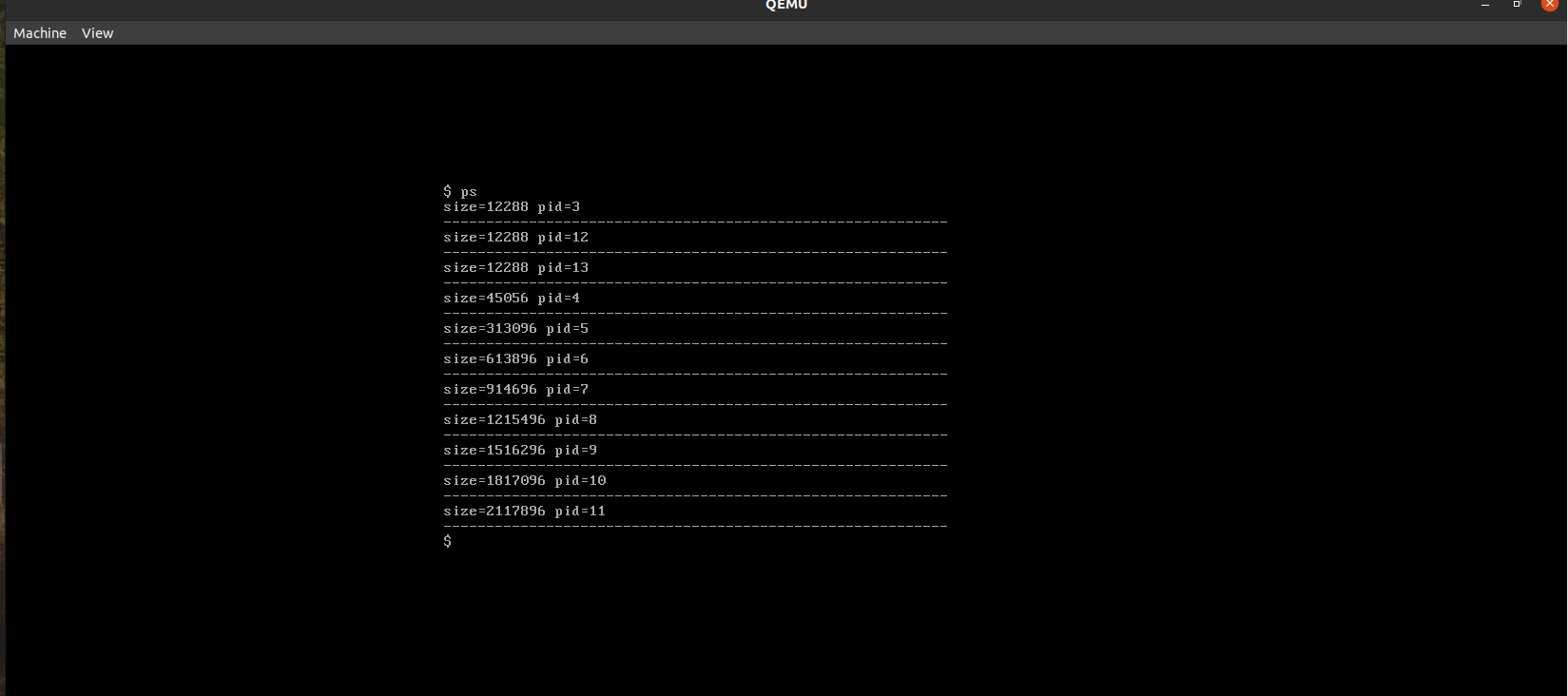


مشکلات ایجاد شده در فرایند انجام پروژه:

1. در سطح سیستم عامل ما نمی توانیم از دستور printf() استفاده کنیم و باید از cprintf() استفاده کنیم
2. ایجادzombie process این مشکل را با زدن یک while(1) ودستور memset() حل کردم

و.....

یک نمونه خروجی:



پایان

با تشکر